COMPUTER NETWORKS

UNIT-I

1.1 Uses of Computer Networks

1.1.1 Business Applications

To keep track of inventories, to monitor production and do the payroll in the Companies that have separate computers. Initially, each of these computers may have worked in isolation from the others. At some point, management may have decided to connect them to be able to extract and correlate information about the entire company. More important thing in this context is sharing physical resources such as printers, scanners, and CD burners, and sharing information. Every large and medium-sized company and many small companies are vitally dependent on computerized information. Most companies have customer records, inventories, accounts receivable, financial statements, tax information, and much more online. Even a small travel agency or three-person law firm is now highly dependent on computer networks for allowing employees to access relevant information and documents instantly.

A second goal of setting up a computer network has to do with people rather than information or even computers. A computer network can provide a powerful communication medium among employees. e-mail is not the only form of improved communication made possible by computer networks, it is easy for two or more people who work far apart to write a report together. Yet another form of computer-assisted communication is videoconferencing. Using this technology, employees at distant locations can hold a meeting, seeing and hearing each other and even writing on a shared virtual blackboard. A third goal for increasingly many companies is doing business electronically with other companies, especially suppliers and customers. A fourth goal that is starting to become more important is doing business with consumers over the Internet. Airlines, bookstores, and music vendors have discovered that many customers like the convenience of shopping from home.

1.1.2 Home Applications

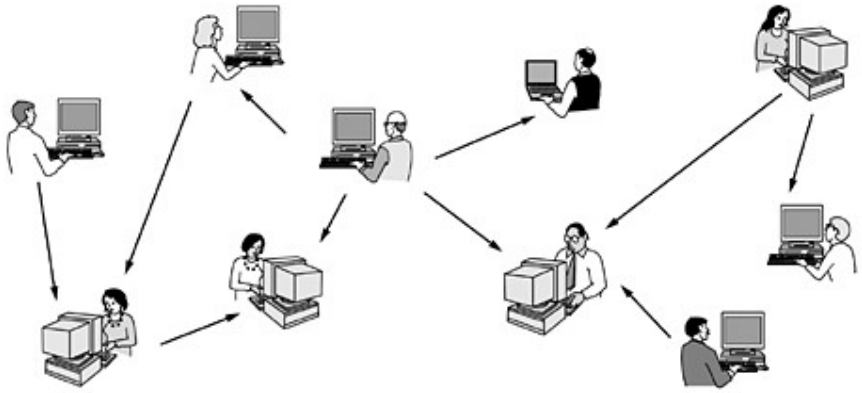
Why do people buy computers for home use?

Initially, for word processing and games, but in recent years that picture has changed radically. Probably the biggest reason now is for Internet access. Some of the more popular uses of the Internet for home users are as follows:

1. Access to remote information.
2. Person-to-person communication.
3. Interactive entertainment.
4. Electronic commerce.

Access to remote information comes in many forms. It can be surfing the World Wide Web for information available includes the arts, business, cooking, government, health, history, hobbies, recreation, science, sports, travel, and many others. Many newspapers have gone on-line and can be personalized.

Another type of **person-to-person communication** often goes by the name of peer-to-peer communication, to distinguish it from the client-server model. In this form, individuals who form a loose group can communicate with others in the group, as shown in the following figure. Every person can, in principle, communicate with one or more other people; there is no fixed division into clients and servers.



Our third category is **entertainment**, which is a huge and growing industry. The killer application here (the one that may drive all the rest) is video on demand. A decade or so hence, it may be possible to select any movie or television program ever made, in any country, and have it displayed on your screen instantly.

Next **game playing**, already we have multi-person real-time simulation games, like hide-and-seek in a virtual dungeon, and flight simulators with the players on one team trying to shoot down the players on the opposing team.

Our fourth category is **electronic commerce** in the broadest sense of the term. Home shopping is already popular and enables users to inspect the on-line catalogs of thousands of companies. Many people already pay their bills, manage their bank accounts, and handle their investments electronically.

1.1.3 Mobile Users

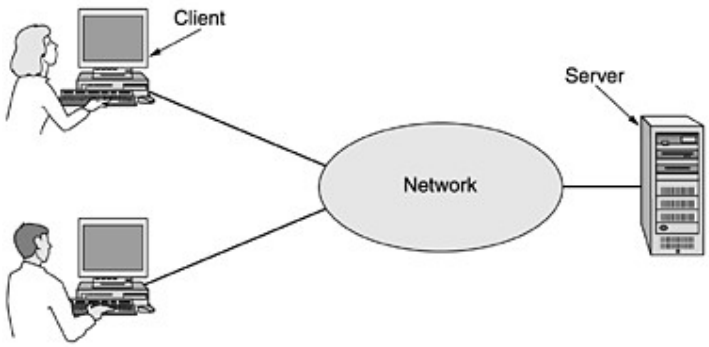
Mobile computers, such as notebook computers and personal digital assistants (PDAs), are one of the fastest growing segments of the computer industry. Many users of these computers have desktop machines back at the office and want to be connected to their home base even when away from home or en route. There is a lot of interest in wireless networks. Wireless networks are also important to the military. Another area in which wireless could save money is utility meter reading. If electricity, gas, water, and other meters in people's homes were to report usage over a wireless network. Wireless smoke detectors could call the fire department instead of making a big noise

1.1.4 Social Issues

The widespread introduction of networking has introduced new social, ethical, and political problems. A popular feature of many networks are newsgroups or bulletin boards whereby people can exchange messages with like-minded individuals. As long as the subjects are restricted to technical topics or hobbies like gardening, not too many problems will arise. The trouble comes when newsgroups are set up on topics that people actually care about, like politics, religion, or sex. Views posted to such groups may be deeply offensive to some people. Messages need not be limited to text. High-resolution color photographs and even short video clips can now easily be transmitted over computer networks. Some people take a live-and-let live view, but others feel that posting certain materials simply unacceptable and must be censored. Different countries have different and conflicting laws in this area. Thus, the debate rages.

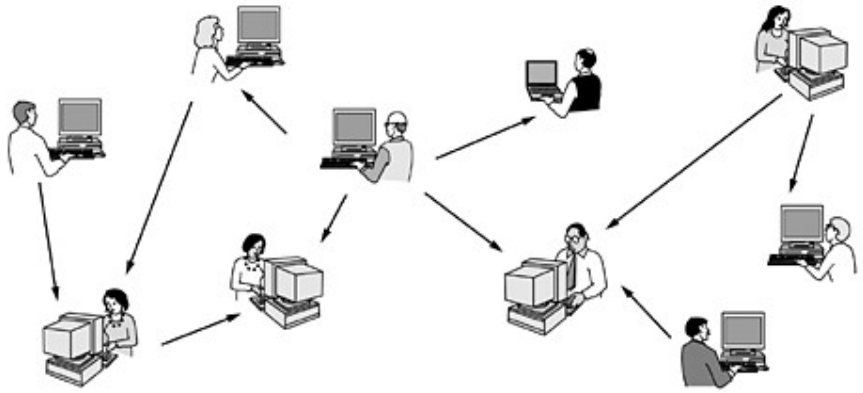
1.2 Client-Server Model:

For smaller companies, all the computers are likely to be in a single office or perhaps a single building, but for larger ones, the computers and employees may be scattered over dozens of offices and plants in many countries. In the simplest of terms, one can imagine a company's information system as consisting of one or more databases and some number of employees who need to access them remotely. In this model, the data are stored on powerful computers called servers. Often these are centrally housed and maintained by a system administrator. In contrast, the employees have simpler machines, called clients, on their desks, with which they access remote data. The client and server machines are connected by a network, as illustrated in the following figure.



1.3 Peer-to-peer communication:

In this form, individuals who form a loose group can communicate with others in the group, as shown in the following figure. Every person can, in principle, communicate with one or more other people; there is no fixed division into clients and servers.



Peer-to-peer communication really hit the big time around 2000 with a service called Napster, which at its peak had over 50 million music fans swapping music, in what was probably the biggest copyright infringement in all of recorded history. The idea was fairly simple. Members registered the music they had on their hard disks in a central database maintained on the Napster server. If a member wanted a song, he checked the database to see who had it and went directly there to get it, by not actually keeping any music on its machines.

1.4 Network Hardware

There are two dimensions of taxonomy stand out as important for computer networks fit: **transmission technology** and **scale.**

1.4.1 Classification based on Transmission Technology

Broadly speaking, there are two types of transmission technology that are in widespread use. They are as follows:

1. Broadcast links.
2. Point-to-point links.

1.4.1.1 Broadcast networks

Broadcast networks have a single communication channel that is shared by all the machines on the network. Short messages, called packets in certain contexts, sent by any machine are received by all the others. An address field within the packet specifies the intended recipient. Upon receiving a packet, a machine checks the address field. If the packet is intended for the receiving machine, that machine processes the packet; if the packet is intended for some other machine, it is just ignored. When a packet is sent to a certain group, it is delivered to all machines subscribing to that group. As a general rule smaller, geographically localized networks tend to use broadcasting, whereas larger networks usually are point-to-point.

1.4.1.2 Point-to-point network

Point-to-point transmission with one sender and one receiver is sometimes called unicasting.

1.4.1.3. Internetwork

Finally, the connection of two or more networks is called an internetwork. The worldwide Internet is a well-known example of an internetwork.

1.4.2 Classification based on Size

Distance is important as a classification metric because different techniques are used at different scales.

Classification of interconnected processors by scale.

Interprocessor distance	Processors located in same	Example
1 m	Square meter	Personal area network
10 m	Room	Local area network
100 m	Building	
1 km	Campus	
10 km	City	Metropolitan area network
100 km	Country	Wide area network
1000 km	Continent	
10,000 km	Planet	The Internet

1.4.2.1 Local Area Networks(LAN)

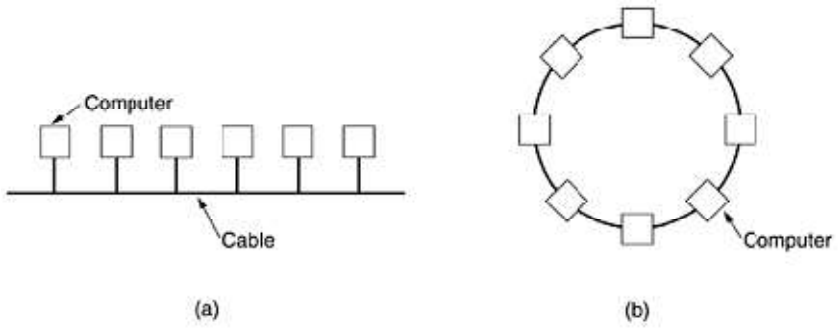
Local area networks, generally called LANs, are privately-owned networks within a single building or campus of up to a few kilometers in size. They are widely used to connect personal computers and workstations in company offices and factories to share resources (e.g., printers) and exchange information. LANs are distinguished from other kinds of networks by three characteristics:

1. their size
2. their transmission technology
3. their topology.

LANs may use a transmission technology consisting of a cable to which all the machines are attached. Traditional LANs run at speeds of 10 Mbps to 100 Mbps, have low delay (microseconds or nanoseconds), and make very few errors. Newer LANs operate at up to 10 Gbps. In this book, we will adhere to tradition and measure line speeds in megabits/sec (1 Mbps is 1,000,000 bits/sec) and

gigabits/sec (1 Gbps is 1,000,000,000 bits/sec). Various topologies are possible for broadcast LANs. Following figure shows two of them.

Two broadcast networks. (a) Bus. (b) Ring.



Bus Topology

In a bus (i.e., a linear cable) network, at any instant at most one machine is the master and is allowed to transmit. An arbitration mechanism is needed to resolve conflicts when two or more machines want to transmit simultaneously. The arbitration mechanism may be centralized or distributed. IEEE 802.3, popularly called Ethernet, for example, is a bus-based broadcast network with decentralized control, usually operating at 10 Mbps to 10 Gbps. Computers on an Ethernet can transmit whenever they want to; if two or more packets collide, each computer just waits a random time and tries again later.

Ring Topology

In a ring, each bit propagates around on its own, not waiting for the rest of the packet to which it belongs. As with all other broadcast systems, some rule is needed for arbitrating simultaneous accesses to the ring. Various methods, such as having the machines take turns, are in use. IEEE 802.5 (the IBM token ring), is a ring-based LAN operating at 4 and 16 Mbps. FDDI is another example of a ring network.

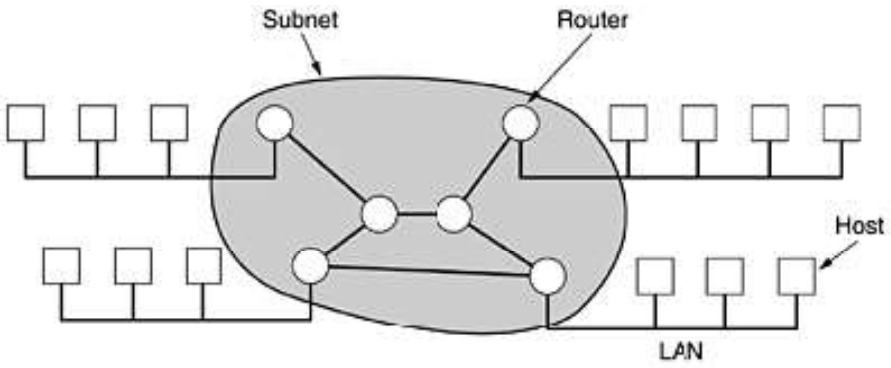
1.4.2.2 Metropolitan Area Networks (MAN)

A metropolitan area network, or MAN, covers a city. The best-known example of a MAN is the cable television network available in many cities. Starting when the Internet attracted a mass audience, the cable TV network operators began to realize that with some changes to the system, they could provide two-way Internet service in unused parts of the spectrum. Cable television is not the only MAN. Recent developments in high-speed wireless Internet access resulted in another MAN, which has been standardized as IEEE 802.16.

1.4.2.3 Wide Area Networks(WAN)

A wide area network, or WAN, spans a large geographical area, often a country or continent. It contains a collection of machines intended for running user (i.e., application) programs. The hosts are connected by a communication subnet, or just subnet for short. The hosts are owned by the customers (e.g., people's personal computers), whereas the communication subnet is typically owned and operated by a telephone company or Internet service provider. The job of the subnet is to carry messages from host to host, just as the telephone system carries words from speaker to listener. In most wide area networks, the subnet consists of two distinct components: transmission lines and switching elements.

Transmission lines move bits between machines. They can be made of copper wire, optical fiber, or even radio links. Switching elements are specialized computers that connect three or more transmission lines. When data arrive on an incoming line, the switching element must choose an outgoing line on which to forward them. These switching computers have been called as router.



When a packet is sent from one router to another via one or more intermediate routers, the packet is received at each intermediate router in its entirety, stored there until the required output line is free, and then forwarded. A subnet organized according to this principle is called a store-and-forward or packet-switched subnet.

1.5 Wireless Networks

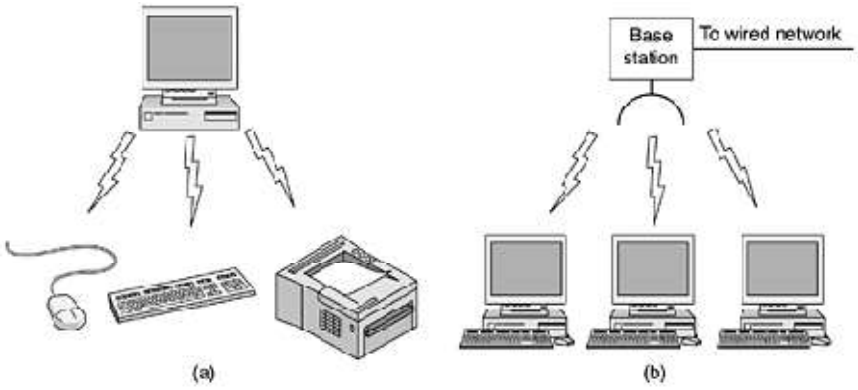
Digital wireless communication was there in 1901 itself, the Italian physicist Guglielmo Marconi demonstrated a ship-to-shore wireless telegraph, using Morse Code (dots and dashes are binary, after all). Modern digital wireless systems have better performance, but the basic idea is the same. To a first approximation, wireless networks can be divided into three main categories:

1. System interconnection.
2. Wireless LANs.
3. Wireless WANs.

1.5.1 System interconnection

System interconnection is all about interconnecting the components of a computer using short-range radio. Almost every computer has a monitor, keyboard, mouse, and printer connected to the main unit by cables. Some companies got together to design a short-range wireless network called Bluetooth to connect these components without wires. Bluetooth also allows digital cameras, headsets, scanners, and other devices to connect to a computer by merely being brought within range. No cables, no driver installation, just put them down, turn them on, and they work. For many people, this ease of operation is a big plus

(a) Bluetooth configuration. (b) Wireless LAN.



1.5.2 Wireless LANs

The next step up in wireless networking are the wireless LANs. These are systems in which every computer has a radio modem and antenna with which it can communicate with other systems. Often there is an antenna on the ceiling that the machines talk to, as shown in figure. Wireless LANs are becoming increasingly common in small offices and homes, where installing Ethernet is considered too much trouble, as well as in older office buildings, company cafeterias, conference rooms, and other places. There is a standard for wireless LANs, called IEEE 802.11, which most systems implement and which is becoming very widespread.

1.5.3 Wireless WANs

The third kind of wireless network is used in wide area systems. The radio network used for cellular telephones is an example of a low-bandwidth wireless system. This system has already gone through three generations.

- The first generation was analog and for voice only.
- The second generation was digital and for voice only.
- The third generation is digital and is for both voice and data.

In a certain sense, cellular wireless networks are like wireless LANs, except that the distances involved are much greater and the bit rates much lower. In addition to these low-speed networks,

high-bandwidth wide area wireless networks are also being developed. Almost all wireless networks hook up to the wired network at some point to provide access to files, databases, and the Internet.

1.5.4 Home Networks

Every device in the home will be capable of communicating with every other device, and all of them will be accessible over the Internet. Many devices are capable of being networked. Some of the more obvious categories (with examples) are as follows:

1. Computers (desktop PC, notebook PC, PDA, shared peripherals).
2. Entertainment (TV, DVD, VCR, camcorder, camera, stereo, MP3).
3. Telecommunications (telephone, mobile telephone, intercom, fax).
4. Appliances (microwave, refrigerator, clock, furnace, airco, lights).
5. Telemetry (utility meter, smoke/burglar alarm, thermostat, babycam).

Home computer networking is already here in a limited way. Many homes already have a device to connect multiple computers to a fast Internet connection. Finally, remote monitoring of the home and its contents is a likely winner. Probably many parents would be willing to spend some money to monitor their sleeping babies on their PDAs when they are eating out, even with a rented teenager in the house. While one can imagine a separate network for each application area, integrating all of them into a single network is probably a better idea. Low price is essential for success. The main application is likely to involve multimedia, Security and reliability will be very important.

Most homes already have six networks installed: electricity, telephone, cable television, water, gas, and sewer. Adding a seventh one during construction is not difficult, but retrofitting existing houses is expensive. Cost favors wireless networking, but security favors wired networking. The problem with wireless is that the radio waves they use are quite good at going through fences.

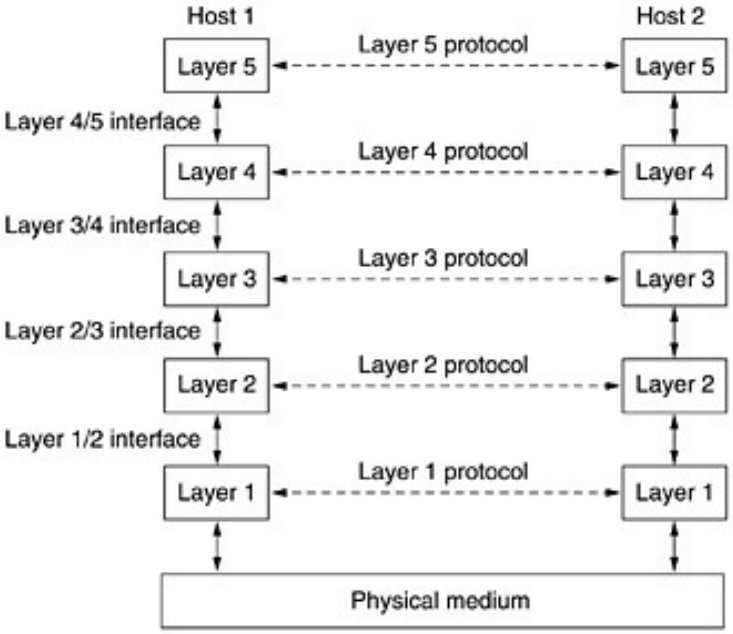
1.5.5 Internetworks

Many networks exist in the world, often with different hardware and software. People connected to one network often want to communicate with people attached to a different one. The fulfillment of this requires that different, and frequently incompatible networks, be connected, sometimes by means of machines called gateways to make the connection and provide the necessary translation, both in terms of hardware and software. A collection of interconnected networks is called an internetwork or internet. Subnet makes the most sense in the context of a wide area network, where it refers to the collection of routers and communication lines owned by the network operator. The combination of a subnet and its hosts forms a network. An internetwork is formed when distinct networks are interconnected.

1.6 Network Software

1.6.1 Protocol Hierarchies

To reduce their design complexity, most networks are organized as a stack of layers or levels, each one built upon the one below it. The number of layers, the name of each layer, the contents of each layer, and the function of each layer differ from network to network. The purpose of each layer is to offer certain services to the higher layers, shielding those layers from the details of how the offered services are actually implemented. Layer n on one machine carries on a conversation with layer n on another machine. The rules and conventions used in this conversation are collectively known as the layer n protocol. A protocol is an agreement between the communicating parties on how communication is to proceed. A five-layer network is illustrated in the following figure.



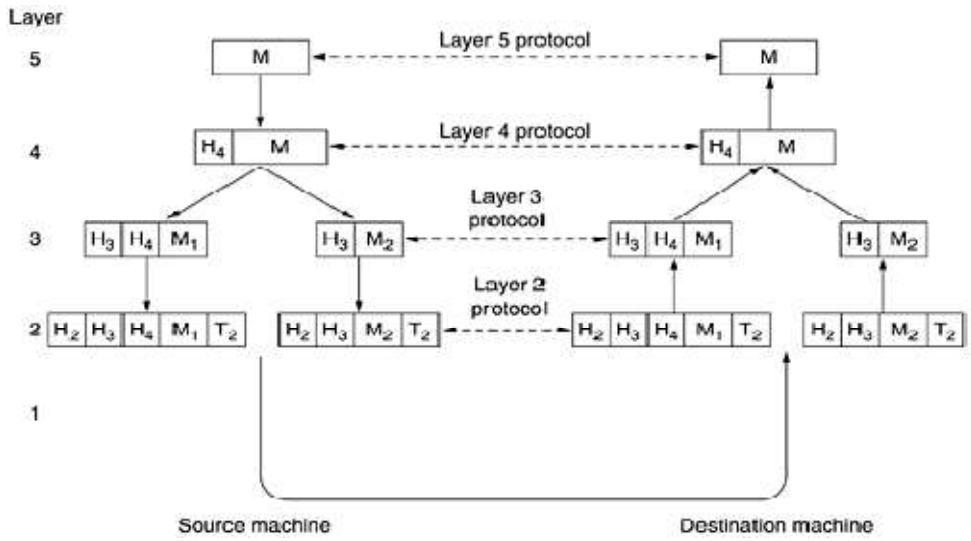
The entities comprising the corresponding layers on different machines are called peers. The peers may be processes, hardware devices, or even human beings. In other words, it is the peers that communicate by using the protocol. In reality, no data are directly transferred from layer n on one machine to layer n on another machine. Instead, each layer passes data and control information to the layer immediately below it, until the lowest layer is reached.

Layer 1 is the physical medium through which actual communication occurs. In the above figure, virtual communication is shown by dotted lines and physical communication by solid lines. Between each pair of adjacent layers is an interface. The interface defines which primitive operations and services the lower layer makes available to the upper one. When network designers decide how many layers to include in a network and what each one should do, one of the most important

considerations is defining clean interfaces between the layers. In addition to minimizing the amount of information that must be passed between layers, clear-cut interfaces also make it simpler to replace the implementation of one layer with a completely different implementation because all that is required of the new implementation is that it offer exactly the same set of services to its upstairs neighbor as the old implementation did. A set of layers and protocols is called a network architecture. A list of protocols used by a certain system, one protocol per layer, is called a protocol stack.

Communication to the top layer of the five-layer networking is shown in the following figure.

Example information flow supporting virtual communication in layer 5.



A message, M, is produced by an application process running in layer 5 and given to layer 4 for transmission. Layer 4 puts a header in front of the message to identify the message and passes the result to layer 3. The header includes control information, such as sequence numbers, to allow layer 4 on the destination machine to deliver messages in the right order if the lower layers do not maintain sequence. In some layers, headers can also contain sizes, times, and other control fields. In many networks, there is no limit to the size of messages transmitted in the layer 4 protocol, but there is nearly always a limit imposed by the layer 3 protocol. Layer 3 must break up the incoming messages into smaller units, packets, prepending a layer 3 header to each packet. In this example, M is split into two parts, M1 and M2.

Layer 3 decides which of the outgoing lines to use and passes the packets to layer 2. Layer 2 adds not only a header to each piece, but also a trailer, and gives the resulting unit to layer 1 for physical transmission. At the receiving machine the message moves upward, from layer to layer, with headers being stripped off as it progresses. The peer process abstraction is crucial to all network design.

Using it, the unmanageable task of designing the complete network can be broken into several smaller, manageable design problems, namely, the design of the individual layers.

1.6.2 Connection-Oriented and Connectionless Services

Layers can offer two different types of services to the layers above them: connection-oriented and connectionless.

1.6.2.1 Connection-oriented service:

Connection-oriented service is modeled after the telephone system. To talk to someone, you pick up the phone, dial the number, talk, and then hang up. To use a connection-oriented network service, the service user first establishes a connection, uses the connection, and then releases the connection. In some cases when a connection is established, the sender, receiver, and subnet conduct a negotiation about parameters to be used, such as maximum message size, quality of service required, and other issues.

1.6.2.2 Connectionless service:

Connectionless service is modeled after the postal system. Each message (letter) carries the full destination address, and each one is routed through the system independent of all the others. When two messages are sent to the same destination, the first one sent will be the first one to arrive. Each service can be characterized by a quality of service. Some services are reliable in the sense that they never lose data.

Reliable connection-oriented service has two minor variations:

1. Message sequences
2. Byte streams.

In the **Message sequences**, the message boundaries are preserved. When two 1024-byte messages are sent, they arrive as two distinct 1024-byte messages, never as one 2048-byte message. In the **Byte streams**, the connection is simply a stream of bytes, with no message boundaries. When 2048 bytes arrive at the receiver, there is no way to tell if they were sent as one 2048-byte message, two 1024-byte messages, or 2048 1-byte messages. If the pages of a book are sent over a network to a phototypesetter as separate messages, it might be important to preserve the message boundaries.

On the other hand, when a user logs into a remote server, a byte stream from the user's computer to the server is all that is needed. All that is needed is a way to send a single message that has a high probability of arrival, but no guarantee. Unreliable (meaning not acknowledged) connectionless service is often called datagram service. In other situations, the convenience of not having to establish a connection to send one short message is desired, but reliability is essential. The following table summarizes the types of services discussed above.

Six different types of service.

	Service	Example
Connection-oriented	Reliable message stream	Sequence of pages
	Reliable byte stream	Remote login
	Unreliable connection	Digitized voice
Connection-less	Unreliable datagram	Electronic junk mail
	Acknowledged datagram	Registered mail
	Request-reply	Database query

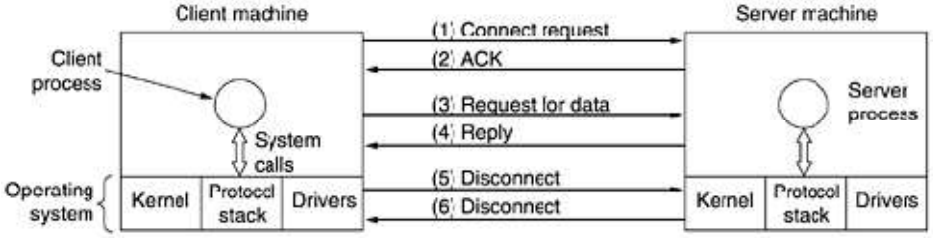
1.6.3 Service Primitives

A service is formally specified by a set of primitives (operations) available to a user process to access the service. These primitives tell the service to perform some action or report on an action taken by a peer entity. The set of primitives available depends on the nature of the service being provided. The primitives for connection-oriented service are different from those of connectionless service. As a minimal example of the service primitives that might be provided to implement a reliable byte stream in a client-server environment, these primitives listed in the following table.

Five service primitives for implementing a simple connection-oriented service.

Primitive	Meaning
LISTEN	Block waiting for an incoming connection
CONNECT	Establish a connection with a waiting peer
RECEIVE	Block waiting for an incoming message
SEND	Send a message to the peer
DISCONNECT	Terminate a connection

Packets sent in a simple client-server interaction on a connection-oriented network.



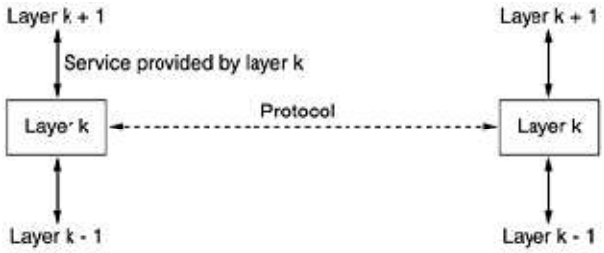
The obvious analogy between this protocol and real life is a customer (client) calling a company's customer service manager. The service manager starts out by being near the telephone in case it rings. Then the client places the call. When the manager picks up the phone, the connection is established. The next step is for the server to execute RECEIVE to prepare to accept the first request. The RECEIVE call blocks the server. Then the client executes SEND to transmit its request (3) followed by the execution of RECEIVE to get the reply.

The arrival of the request packet at the server machine unblocks the server process so it can process the request. After it has done the work, it uses SEND to return the answer to the client (4). The arrival of this packet unblocks the client, which can now inspect the answer. If the client has additional requests, it can make them now. If it is done, it can use DISCONNECT to terminate the connection. Usually, an initial DISCONNECT is a blocking call, suspending the client and sending a packet to the server saying that the connection is no longer needed (5). When the server gets the packet, it also issues a DISCONNECT of its own, acknowledging the client and releasing the connection. When the server's packet (6) gets back to the client machine, the client process is released and the connection is broken. In a nutshell, this is how connection-oriented communication works. Of course, life is not so simple. Many things can go wrong here. The timing can be wrong.

1.6.4 The Relationship of Services to Protocols

Services and protocols are distinct concepts, although they are frequently confused. This distinction is so important, however, that we emphasize it again here. A service is a set of primitives (operations) that a layer provides to the layer above it. The service defines what operations the layer is prepared to perform on behalf of its users, but it says nothing at all about how these operations are implemented. A service relates to an interface between two layers, with the lower layer being the service provider and the upper layer being the service user. A protocol, in contrast, is a set of rules governing the format and meaning of the packets, or messages that are exchanged by the peer entities within a layer. Entities use protocols to implement their service definitions. They are free to change their protocols at will, provided they do not change the service visible to their users. In this way, the service and the protocol are completely decoupled.

The relationship between a service and a protocol.

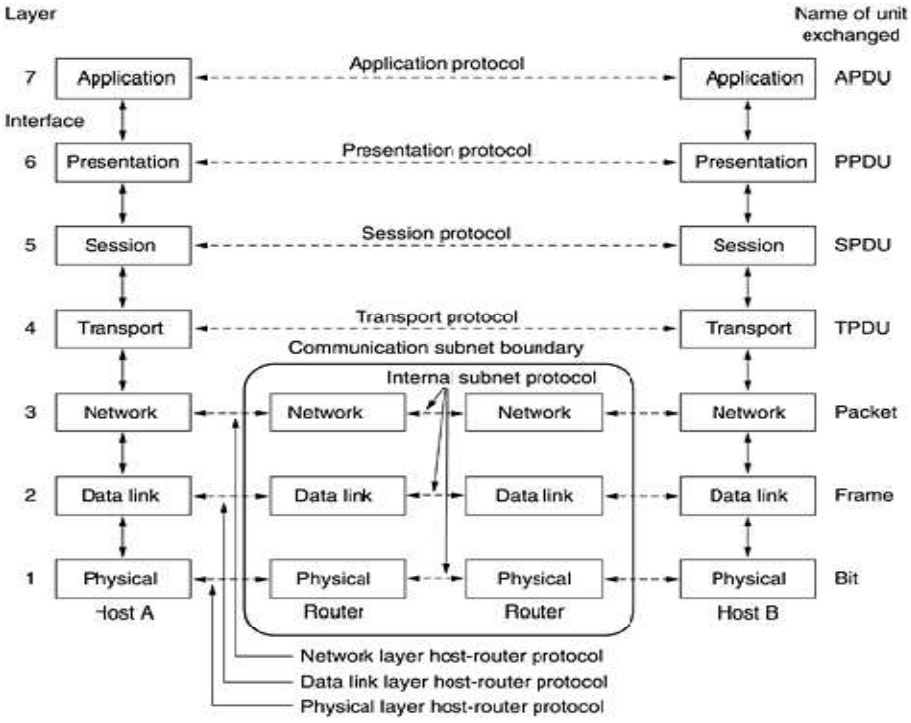


1.7 Reference Models

1.7.1 The OSI Reference Model

This model is based on a proposal developed by the International Standards Organization (ISO) as a first step toward international standardization of the protocols used in the various layers

The OSI reference model.



The OSI model has seven layers. The principles that were applied to arrive at the seven layers can be briefly summarized as follows:

1. A layer should be created where a different abstraction is needed.
2. Each layer should perform a well-defined function.
3. The function of each layer should be chosen with an eye toward defining internationally standardized 37 protocols.
4. The layer boundaries should be chosen to minimize the information flow across the interfaces.
5. The number of layers should be large enough that distinct functions need not be thrown together in the same layer out of necessity and small enough that the architecture does not become unwieldy.

1.7.1 Physical Layer

The physical layer coordinates the functions required to carry a bit stream over a physical medium. It deals with the mechanical and electrical specifications of the interface and transmission medium.

The physical layer is also concerned with the following:

- **Physical characteristics of interfaces and medium.** The physical layer defines the characteristics of the interface between the devices and the transmission medium
- **Representation of bits.** The physical layer data consists of a stream of bits (sequence of Os or 1s) with no interpretation. To be transmitted, bits must be encoded into signals--electrical or optical.
- **Data rate.** The transmission rate-the number of bits sent each second-is also defined by the physical layer.
- **Synchronization of bits.** The sender and receiver not only must use the same bit rate but also must be synchronized at the bit level.
- **Line configuration.** The physical layer is concerned with the connection of devices to the media. In a point-to-point configuration, two devices are connected through a dedicated link. In a multipoint configuration, a link is shared among several devices.
- **Physical topology.** The physical topology defines how devices are connected to make a network. Devices can be connected by using a various topologies.
- **Transmission mode.** The physical layer also defines the direction of transmission between two devices: simplex(one-way communication), half-duplex(two devices can send and receive, but not at the same time), or full-duplex(two devices can send and receive at the same time).

1.7.1 Data Link Layer

The data link layer transforms the physical layer, a raw transmission facility, to a reliable link. It makes the physical layer appear error-free to the upper layer (network layer).

Other responsibilities of the data link layer include the following:

- **Framing**. The data link layer divides the stream of bits received from the network layer into manageable data units called frames.
- **Physical addressing.** If frames are to be distributed to different systems on the network, the data link layer adds a header to the frame to define the sender and/or receiver of the frame. If the frame is intended for a system outside the sender's network, the receiver address is the address of the device that connects the network to the next one.
- **Flow control.** If the rate at which the data are absorbed by the receiver is less than the rate at which data are produced in the sender, the data link layer imposes a flow control mechanism to avoid overwhelming the receiver.
- **Error control.** The data link layer adds reliability to the physical layer by adding mechanisms to detect and retransmit damaged or lost frames. It also uses a mechanism to recognize duplicate frames. Error control is normally achieved through a trailer added to the end of the frame.
- **Access control.** When two or more devices are connected to the same link, data link layer protocols are necessary to determine which device has control over the link at any given time.

1.7.3 Network Layer

The network layer is responsible for the source-to-destination delivery of a packet, possibly across multiple networks (links).

Other responsibilities of the network layer include the following:

- **Logical addressing.** The physical addressing implemented by the data link layer handles the addressing problem locally. If a packet passes the network boundary, we need another addressing system to help distinguish the source and destination systems. The network layer adds a header to the packet coming from the upper layer that, among other things, includes the logical addresses of the sender and receiver.
- **Routing.** When independent networks or links are connected to create *intemetworks* (network of networks) or a large network, the connecting devices (called *routers* or *switches)* route or switch the packets to their final destination. One of the functions of the network layer is to provide this mechanism.

1.7.4 Transport Layer

The transport layer is responsible for process-to-process delivery of the entire message.
Other responsibilities of the transport layer include the following:

- **Service-point addressing.** Computers often run several programs at the same time. For this reason, source-to-destination delivery means delivery not only from one computer to the next but also from a specific process (running program) on one computer to a specific process (running program) on the other. The transport layer header must therefore include a type of address called a *service-point address* (or port address). The network layer gets each packet to the correct computer; the transport layer gets the entire message to the correct process on that computer.
- **Segmentation and reassembly.** A message is divided into transmittable segments, with each segment containing a sequence number.
- **Connection control.** The transport layer can be either connectionless or connection oriented. A connectionless transport layer treats each segment as an independent packet and delivers it to the transport layer at the destination machine. A connection oriented transport layer makes a connection with the transport layer at the destination machine first before delivering the packets. After all the data are transferred, the connection is terminated.
- **Flow control.** Like the data link layer, the transport layer is responsible for flow control. However, flow control at this layer is performed end to end rather than across a single link.
- **Error control.** Like the data link layer, the transport layer is responsible for error control. However, error control at this layer is performed process-to process rather than across a single link. The sending transport layer makes sure that the entire message arrives at the receiving transport layer without error (damage, loss, or duplication). Error correction is usually achieved through retransmission.

1.7.5 Session Layer

The services provided by the first three layers (physical, data link, and network) are not sufficient for some processes. The session layer is the network *dialog controller.* It establishes, maintains, and synchronizes the interaction among communicating systems.

Specific responsibilities of the session layer include the following:

- **Dialog control.** The session layer allows two systems to enter into a dialog. It allows the communication between two processes to take place in either half-duplex (one way at a time) or full-duplex (two ways at a time) mode.
- **Synchronization.** The session layer allows a process to add checkpoints, or synchronization points, to a stream of data. For example, if a system is sending a file of 2000 pages, it is advisable to insert checkpoints after every 10.

1.7.6 Presentation Layer

The presentation layer is concerned with the syntax and semantics of the information exchanged between two systems.

Specific responsibilities of the presentation layer include the following:

- **Translation.** The processes (running programs) in two systems are usually exchanging information in the form of character strings, numbers, and so on. The information must be changed to bit streams before being transmitted.
- **Encryption.** To carry sensitive information, a system must be able to ensure privacy. Encryption means that the sender transforms the original information to another form and sends the resulting message out over the network. Decryption reverses the original process to transform the message back to its original form.
- **Compression.** Data compression reduces the number of bits contained in the information. Data compression becomes particularly important in the transmission of multimedia such as text, audio, and video.

1.7.7 Application Layer

The application layer enables the user, whether human or software, to access the network. It provides user interfaces and support for services.

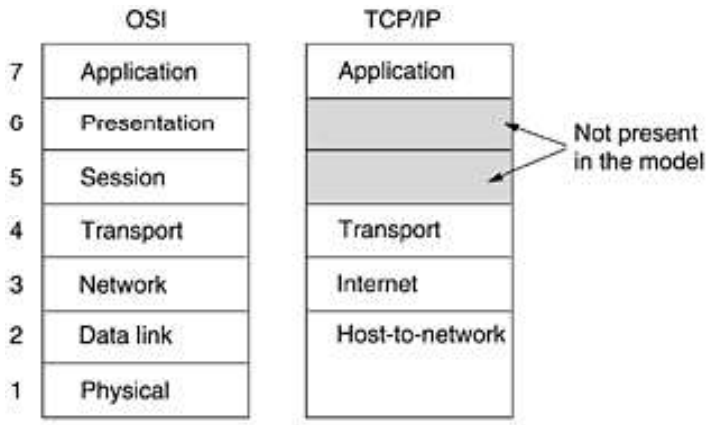
Specific services provided by the application layer include the following:

- **Network virtual terminal.** A network virtual terminal is a software version of a physical terminal, and it allows a user to log on to a remote host. To do so, the application creates a software emulation of a terminal at the remote host. The user's computer talks to the software terminal which, in turn, talks to the host, and vice versa. The remote host believes it is communicating with one of its own terminals and allows the user to log on.

- **File transfer, access, and management.** This application allows a user to access files in a remote host (to make changes or read data), to retrieve files from a remote computer for use in the local computer, and to manage or control files in a remote computer locally.
- **Mail services.** This application provides the basis for e-mail forwarding and storage.
- **Directory services.** This application provides distributed database sources and access for global information about various objects and services.

1.8. The TCP/IP Reference Model

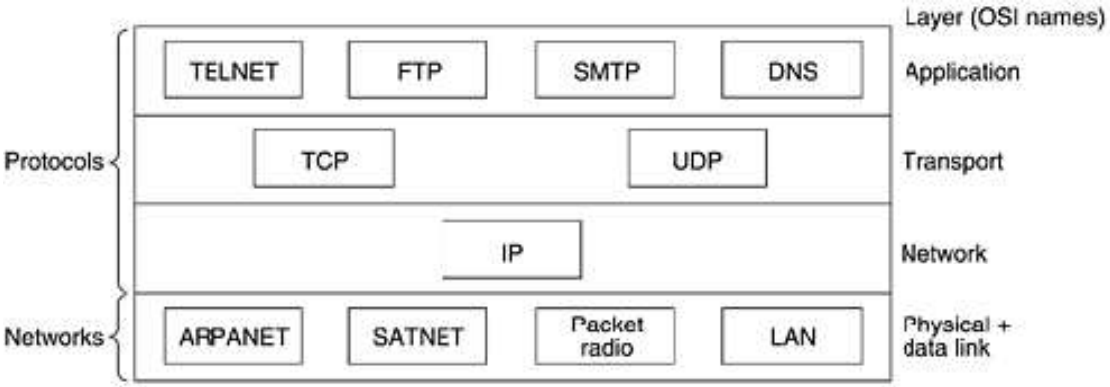
The ARPANET was a research network sponsored by the DoD (U.S. Department of Defense). It eventually connected hundreds of universities and government installations, using leased telephone lines. When satellite and radio networks were added later, the existing protocols had trouble interworking with them, so a new reference architecture was needed. Thus, the ability to connect multiple networks in a seamless way was one of the major design goals from the very beginning. This architecture later became known as the TCP/IP Reference Model,



1.8.1 The Internet Layer

- Its job is to permit hosts to inject packets into any network and have them travel independently to the destination. They may even arrive in a different order than they were sent, in which case it is the job of higher layers to rearrange them, if in-order delivery is desired.
- The internet layer defines an official packet format and protocol called IP (Internet Protocol).
- Packet routing is clearly the major issue here, as is avoiding congestion.

Protocols and networks in the TCP/IP model initially.



1.8.2 The Transport Layer

The layer above the internet layer in the TCP/IP model is now usually called the transport layer.

- It is designed to allow peer entities on the source and destination hosts to carry on a conversation, just as in the OSI transport layer.
- Two end-to-end transport protocols have been defined here. The first one, TCP (Transmission Control Protocol), is a reliable connection-oriented protocol that allows a byte stream originating on one machine to be delivered without error on any other machine in the internet. It fragments the incoming byte stream into discrete messages and passes each one on to the internet layer. At the destination, the receiving TCP process reassembles the received messages into the output stream. TCP also handles flow control to make sure a fast sender cannot swamp a slow receiver with more messages than it can handle.
- The second protocol in this layer, UDP (User Datagram Protocol), is an unreliable, connectionless protocol for applications that do not want TCP's sequencing or flow control and wish to provide their own.

1.8.3 The Application Layer

On top of the transport layer is the application layer. It contains all the higher-level protocols. The early ones included virtual terminal (TELNET), file transfer (FTP), and electronic mail (SMTP)

- The file transfer protocol provides a way to move data efficiently from one machine to another.
- Electronic mail was originally just a kind of file transfer, but later a specialized protocol (SMTP) was developed for it.

- Many other protocols have been added to these over the years: the Domain Name System (DNS) for mapping host names onto their network addresses.
- NNTP, the protocol for moving USENET news articles around.
- HTTP, the protocol for fetching pages on the World Wide Web, and many others.

1.8.4 The Host-to-Network Layer

- The TCP/IP reference model does not really say much about what happens here, except to point out that the host has to connect to the network using some protocol so it can send IP packets to it. This protocol is not defined and varies from host to host and network to network.

The Physical Layer

2.1 Guided Transmission Media

The purpose of the physical layer is to transport a raw bit stream from one machine to another. Various physical media can be used for the actual transmission. Each one has its own importance in terms of bandwidth, delay, cost, and ease of installation and maintenance.

Media are roughly grouped into

1. **Guided media**, such as copper wire and fiber optics.
2. **Unguided media**, such as radio and lasers through the air.

2.1.1 Magnetic Media

One of the most common ways to transport data from one computer to another is to write them onto magnetic tape or removable media (e.g., recordable DVDs), physically transport the tape or disks to the destination machine, and read them back in again. A simple calculation will make this point clear. An industry standard Ultrium tape can hold 200 gigabytes. A box 60 x 60 x 60 cm can hold about 1000 of these tapes, for a total capacity of 200 terabytes, or 1600 terabits (1.6 petabits).

A box of tapes can be delivered manually by road transportation. The effective bandwidth of this transmission is 1600 terabits/86,400 sec, or 19 Gbps. If the destination is only an hour away by road, the bandwidth is increased to over 400 Gbps. The cost of an Ultrium tape is around $40 when bought in bulk. A tape can be reused at least ten times, so the tape cost is maybe $4000 per box per usage.

2.1.2 Twisted Pair

One of the oldest and still most common transmission media is twisted pair. A twisted pair consists of two insulated copper wires, typically about 1 mm thick. The wires are twisted together in a helical form, just like a DNA molecule. Twisting is done because two parallel wires constitute a fine antenna. When the wires are twisted, the waves from different twists cancel out, so the wire radiates

less effectively. The most common application of the twisted pair is the telephone system. Nearly all telephones are connected to the telephone company (telco) office by a twisted pair. Twisted pairs can run several kilometers without amplification, but for longer distances, repeaters are needed. Twisted pairs can be used for transmitting either analog or digital signals. The bandwidth depends on the thickness of the wire and the distance traveled, but several megabits/sec can be achieved for a few kilometers in many cases. Due to their adequate performance and low cost, twisted pairs are widely used and are likely to remain so for years to come.

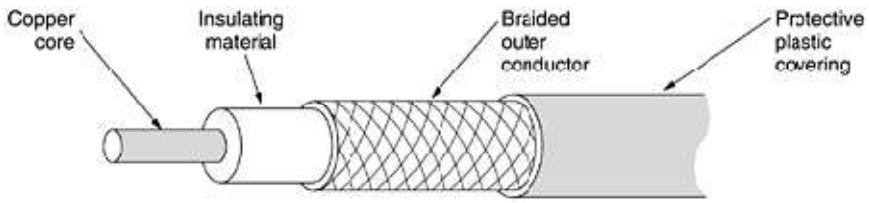
Twisted pair cabling comes in several varieties, two of which are important for computer networks. **Category 3** twisted pairs consist of two insulated wires gently twisted together. Four such pairs are typically grouped in a plastic sheath to protect the wires and keep them together. Starting around 1988, the more advanced **category 5** twisted pairs were introduced. They are similar to category 3 pairs, but with more twists per centimeter, which results in less crosstalk and a better-quality signal over longer distances, making them more suitable for high-speed computer communication. Up-and-coming categories are 6 and 7, which are capable of handling signals with bandwidths of 250 MHz and 600 MHz, respectively. All of these wiring types are often referred to as UTP (Unshielded Twisted Pair), Twisted pair cabling is illustrated in the following figure.

(a) Category 3 UTP. (b) Category 5 UTP.

2.1.3 Coaxial Cable

Another common transmission medium is the coaxial cable (known to its many friends as just "coax" and pronounced "co-ax"). It has better shielding than twisted pairs, so it can span longer distances at higher speeds. Two kinds of coaxial cable are widely used. One kind, 50-ohm cable, is commonly used when it is intended for digital transmission from the start. The other kind, 75-ohm cable, is commonly used for analog transmission and cable television but is becoming more important with the advent of Internet over cable. A coaxial cable consists of a stiff copper wire as the core, surrounded by an insulating material. The insulator is encased by a cylindrical conductor, often as a closely-woven braided mesh. The outer conductor is covered in a protective plastic sheath. A cutaway view of a coaxial cable is shown in following figure.

A coaxial cable.



The construction and shielding of the coaxial cable give it a good combination of high bandwidth and excellent noise immunity. Coax is still widely used for cable television and metropolitan area networks, however.

2.1.4 Fiber Optics

Fiber optics transmission technology works using optical transmission system.

An optical transmission system has three key components:

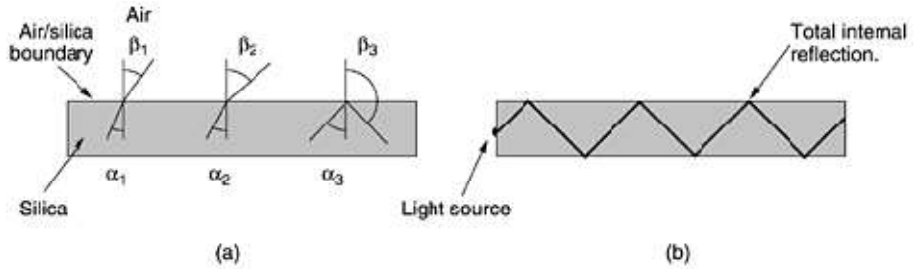
1. the light source
2. the transmission medium
3. the detector

Conventionally, a pulse of light indicates a 1 bit and the absence of light indicates a 0 bit. The transmission medium is an ultra-thin fiber of glass. The detector generates an electrical pulse when light falls on it. By attaching a light source to one end of an optical fiber and a detector to the other, we have a unidirectional data transmission system that accepts an electrical signal, converts and transmits it by light pulses, and then reconverts the output to an electrical signal at the receiving end.

This transmission system would leak light and be useless in practice except for an interesting principle of physics.

When a light ray passes from one medium to another, for example, from fused silica to air, the ray is refracted (bent) at the silica/air boundary, as shown in the following figure(a). Here we see a light ray incident on the boundary at an angle a1 emerging at an angle b1. The amount of refraction depends on the properties of the two media (in particular, their indices of refraction). For angles of incidence above a certain critical value, the light is refracted back into the silica; none of it escapes into the air. Thus, a light ray incident at or above the critical angle is trapped inside the fiber, and can propagate for many kilometers with virtually no loss (figure (b)).

(a) Three examples of a light ray from inside a silica fiber impinging on the air/silica boundary at different angles. (b) Light trapped by total internal reflection.



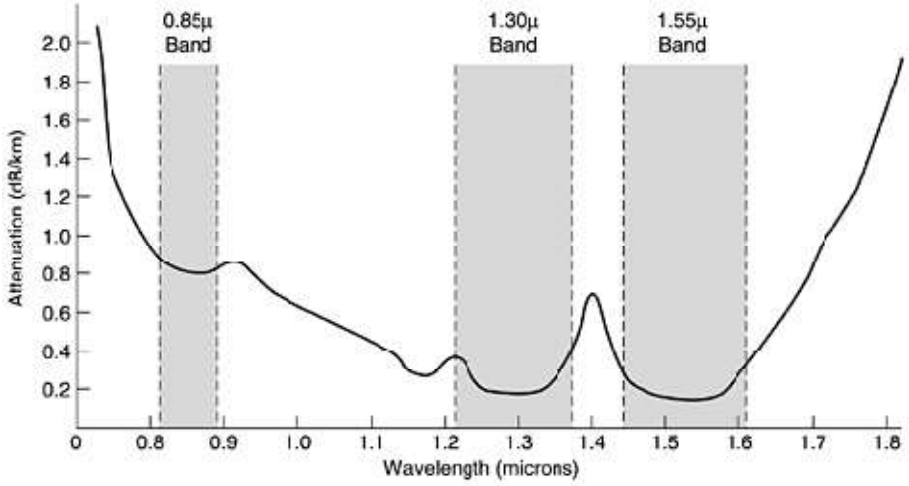
The sketch of figure (b) shows only one trapped ray, but since any light ray incident on the boundary above the critical angle will be reflected internally, many different rays will be bouncing around at different angles. Each ray is said to have a different mode, so a fiber having this property is called a multimode fiber. If the fiber's diameter is reduced to a few wavelengths of light, the fiber acts like a wave guide, and the light can propagate only in a straight line, without bouncing, yielding a single-

mode fiber. Single-mode fibers are more expensive but are widely used for longer distances. Currently available single-mode fibers can transmit data at 50 Gbps for 100 km without amplification.

2.1.4.1 Transmission of Light through Fiber

Optical fibers are made of glass, which, in turn, is made from sand, an inexpensive raw material available in unlimited amounts. The attenuation of light through glass depends on the wavelength of the light (as well as on some physical properties of the glass). For the kind of glass used in fibers, the attenuation is shown in figure in decibels per linear kilometer of fiber.

Attenuation of light through fiber in the infrared region.



The attenuation in decibels is given by the formula

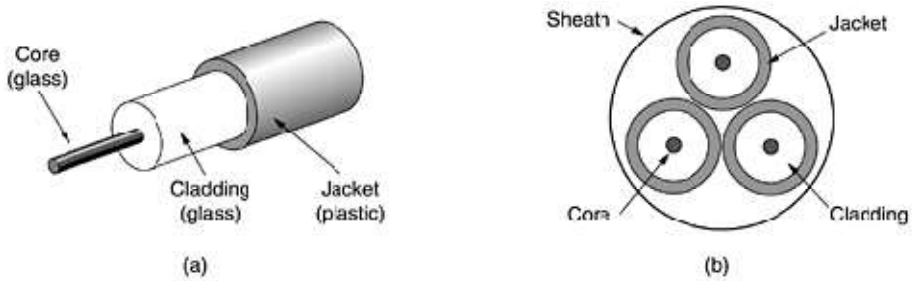
$$\text{Attenuation in decibels} = 10 \log_{10} \frac{\text{transmitted power}}{\text{received power}}$$

Three wavelength bands are used for optical communication. They are centered at 0.85, 1.30, and 1.55 microns, respectively. The last two have good attenuation properties (less than 5 percent loss per kilometer).

2.1.4.2 Fiber Cables

Fiber optic cables are similar to coax, except without the braid. The following figure shows a single fiber viewed from the side. At the center is the glass core through which the light propagates. In multimode fibers, the core is typically 50 microns in diameter, about the thickness of a human hair. In single-mode fibers, the core is 8 to 10 microns.

(a) Side view of a single fiber. (b) End view of a sheath with three fibers.



The core is surrounded by a glass cladding with a lower index of refraction than the core, to keep all the light in the core. Next comes a thin plastic jacket to protect the cladding. Fibers are typically grouped in bundles, protected by an outer sheath. Figure(b) shows a sheath with three fibers. Terrestrial fiber sheaths are normally laid in the ground within a meter of the surface. In deep water, they just lie on the bottom. Fibers can be connected in three different ways.

1. First, they can terminate in connectors and be plugged into fiber sockets. Connectors lose about 10 to 20 percent of the light, but they make it easy to reconfigure systems.
2. Second, they can be spliced mechanically. Mechanical splices just lay the two carefully-cut ends next to each other in a special sleeve and clamp them in place.
3. Third, two pieces of fiber can be fused (melted) to form a solid connection. A fusion splice is almost as good as a single drawn fiber, but even here, a small amount of attenuation occurs.

Two kinds of light sources are typically used to do the signaling,

1. LEDs (Light Emitting Diodes)
2. Semiconductor lasers.

They have different properties, as shown in the following figure.

A comparison of semiconductor diodes and LEDs as light sources.

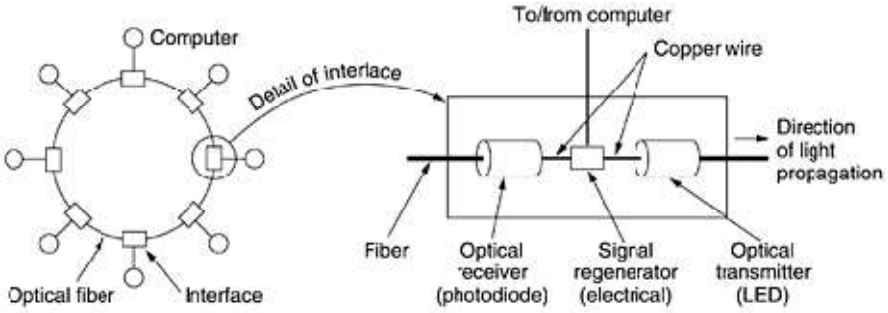
Item	LED	Semiconductor laser
Data rate	Low	High
Fiber type	Multimoce	Multimode or single mode
Distance	Short	Long
Lifetime	Long life	Short life
Temperature sensitivity	Minor	Substantial
Cost	Low cost	Expensive

The receiving end of an optical fiber consists of a photodiode, which gives off an electrical pulse when struck by light. The typical response time of a photodiode is 1 nsec, which limits data rates to about 1 Gbps.

2.1.4.3 Fiber Optic Networks

Fiber optics can be used for LANs as well as for long-haul transmission, although tapping into it is more complex than connecting to an Ethernet. The interface at each computer passes the light pulse stream through to the next link and also serves as a T junction to allow the computer to send and accept messages.

fiber optic ring with active repeaters.



Two types of interfaces are used.

A **passive interface** consists of two taps fused onto the main fiber. One tap has an LED or laser diode at the end of it (for transmitting), and the other has a photodiode (for receiving). The tap itself is completely passive and is thus extremely reliable because a broken LED or photodiode does not break the ring. It just takes one computer off-line.

The other interface type, shown in above figure is the **active repeater**. The incoming light is converted to an electrical signal, regenerated to full strength if it has been weakened, and retransmitted as light. The interface with the computer is an ordinary copper wire that comes into the signal regenerator. Purely optical repeaters are now being used, too. These devices do not require the optical to electrical to optical conversions, which means they can operate at extremely high bandwidths.

2.1.5 Comparison of Fiber Optics and Copper Wire

It is instructive to compare fiber to copper.

Fiber has many advantages. It can handle much higher bandwidths than copper. This alone would require its use in high-end networks. Due to the low attenuation, repeaters are needed only about every 50 km on long lines, versus about every 5 km for copper, a substantial cost saving. Fiber also has the advantage of not being affected by power surges, electromagnetic interference, or power

failures. Nor is it affected by corrosive chemicals in the air, making it ideal for harsh factory environments. Telephone companies like fiber for a different reason: it is thin and lightweight.

Removing all the copper and replacing it by fiber empties the ducts, and the copper has excellent resale value to copper refiners who see it as very high grade ore. Fiber is much lighter than copper. One thousand twisted pairs 1 km long weigh 8000 kg. Two fibers have more capacity and weigh only 100 kg, which greatly reduces the need for expensive mechanical support systems that must be maintained. For new routes, fiber wins hands down due to its much lower installation cost. Finally, fibers do not leak light and are quite difficult to tap. These properties gives fiber excellent security against potential wire tappers.

On the downside, fiber is a less familiar technology requiring skills not all engineers have, and fibers can be damaged easily by being bent too much. Since optical transmission is inherently unidirectional, two-way communication requires either two fibers or two frequency bands on one fiber. Finally, fiber interfaces cost more than electrical interfaces. The future of all fixed data communication for distances of more than a few meters is clearly with fiber.

2.2 Wireless Transmission

Wireless has advantages for even fixed devices in some circumstances. For example, if running a fiber to a building is difficult due to the terrain (mountains, jungles, swamps, etc.), wireless may be better.

2.2.1 The Electromagnetic Spectrum

When electrons move, they create electromagnetic waves that can propagate through space. These waves were predicted by the British physicist James Clerk Maxwell in 1865 and first observed by the German physicist Heinrich Hertz in 1887. When an antenna of the appropriate size is attached to an electrical circuit, the electromagnetic waves can be broadcast efficiently and received by a receiver some distance away. In vacuum, all electromagnetic waves travel at the same speed, no matter what their frequency is. This speed, usually called the speed of light, c, is approximately 3 x 108 m/sec, or about 1 foot (30 cm) per nanosecond. In copper or fiber the speed slows to about 2/3 of this value and becomes slightly frequency dependent. The fundamental relation between f, λ , and c (in vacuum) is,

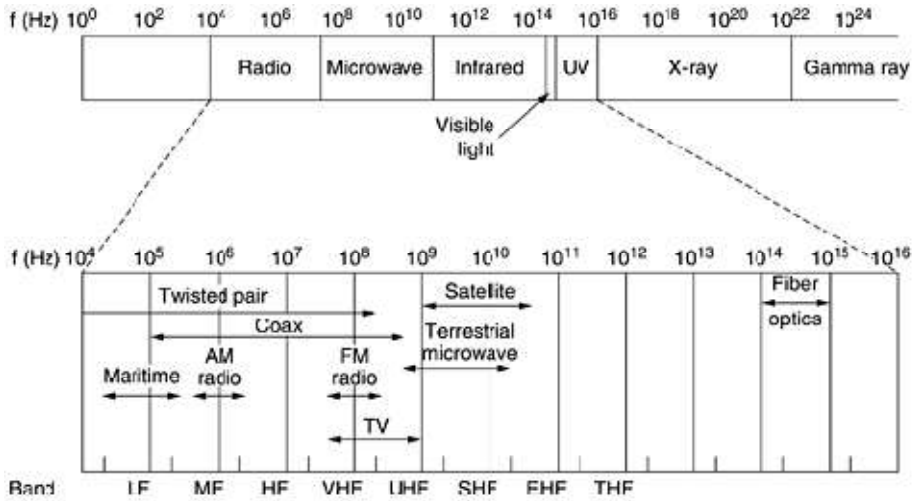
$$\lambda f = c$$

Since c is a constant, if we know f, we can find λ , and vice versa. As a rule of thumb, when λ is in meters and f is in MHz. The electromagnetic spectrum is shown in the following figure. The radio, microwave, infrared, and visible light portions of the spectrum can all be used for transmitting information by modulating the amplitude, frequency, or phase of the waves. Ultraviolet light, X-rays, and gamma rays would be even better, due to their higher frequencies, but they are hard to produce

and modulate, do not propagate well through buildings, and are dangerous to living things. The bands listed at the bottom of the spectrum are the official ITU names and are based on the wavelengths, so the LF band goes from 1 km to 10 km (approximately 30 kHz to 300 kHz). The terms LF, MF, and HF refer to low, medium, and high frequency, respectively.

Clearly, when the names were assigned, nobody expected to go above 10 MHz, so the higher bands were later named the Very, Ultra, Super, Extremely, and Tremendously High Frequency bands. Beyond that there are no names, but Incredibly, Astonishingly, and Prodigiously high frequency (IHF, AHF, and PHF) would sound nice.

The electromagnetic spectrum and its uses for communication.



The amount of information that an electromagnetic wave can carry is related to its bandwidth. The wider the band, the higher the data rate. A wide band is used, with two variations.

1. **Frequency hopping spread spectrum:** The transmitter hops from frequency to frequency hundreds of times per second. It is popular for military communication because it makes transmissions hard to detect and next to impossible to jam.
2. **Direct sequence spread spectrum:** which spreads the signal over a wide frequency band, is also gaining popularity in the commercial world.

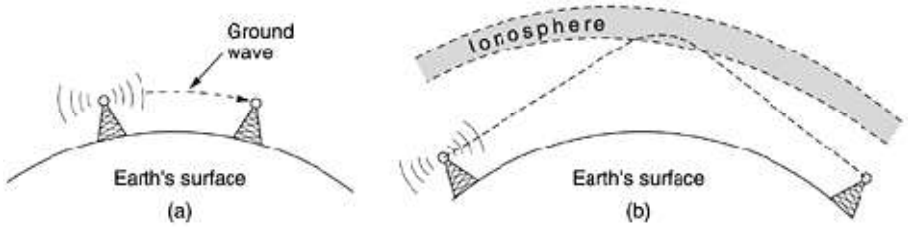
2.2.2 Radio Transmission

Radio waves are easy to generate, can travel long distances, and can penetrate buildings easily, so they are widely used for communication, both indoors and outdoors. Radio waves also are omni directional, meaning that they travel in all directions from the source, so the transmitter and receiver do not have to be carefully aligned physically. The properties of radio waves are frequency dependent. At low frequencies, radio waves pass through obstacles well, but the power falls off

sharply with distance from the source, roughly as 1/r2 in air. At high frequencies, radio waves tend to travel in straight lines and bounce off obstacles. They are also absorbed by rain.

At all frequencies, radio waves are subject to interference from motors and other electrical equipment. Due to radio's ability to travel long distances, interference between users is a problem. For this reason, all governments tightly license the use of radio transmitters. In the VLF, LF, and MF bands, radio waves follow the ground, as illustrated in the following figure. These waves can be detected for perhaps 1000 km at the lower frequencies, less at the higher ones. AM radio broadcasting uses the MF band, which is why the ground waves from Boston AM radio stations cannot be heard easily in New York. Radio waves in these bands pass through buildings easily, which is why portable radios work indoors. The main problem with using these bands for data communication is their low bandwidth.

(a) In the VLF, LF, and MF bands, radio waves follow the curvature of the earth. (b) In the HF band, they bounce off the ionosphere.



In the HF and VHF bands, the ground waves tend to be absorbed by the earth. The waves that reach the ionosphere, a layer of charged particles circling the earth at a height of 100 to 500 km, are refracted by it and sent back to earth, as shown in the above figure. Under certain atmospheric conditions, the signals can bounce several times. Amateur radio operators (hams) use these bands to talk long distance. The military also communicate in the HF and VHF bands.

2.2.3 Microwave Transmission

Above 100 MHz, the waves travel in nearly straight lines and can therefore be narrowly focused. Concentrating all the energy into a small beam by means of a parabolic antenna gives a much higher signal-to-noise ratio, but the transmitting and receiving antennas must be accurately aligned with each other. This directionality allows multiple transmitters lined up in a row to communicate with multiple receivers in a row without interference, provided some minimum spacing rules are observed.

Before fiber optics, for decades these microwaves formed the heart of the long-distance telephone transmission system. Since the microwaves travel in a straight line, if the towers are too far apart, the earth will get in the way, repeaters are needed periodically. The higher the towers are, the farther apart they can be. For 100-meter-high towers, repeaters can be spaced 80 km apart. Unlike

radio waves at lower frequencies, microwaves do not pass through buildings well even though the beam may be well focused at the transmitter, there is still some divergence in space. These waves are only a few centimeters long and are absorbed by rain. Microwave communication is so widely used for long-distance telephone communication, mobile phones, television distribution, and other uses that a severe shortage of spectrum has developed.

2.2.4 The Politics of the Electromagnetic Spectrum

There are national and international agreements about who gets to use which frequencies. Since everyone wants a higher data rate, everyone wants more spectrum. Worldwide, an agency of ITU-R (WARC) tries to coordinate this allocation. Even when a piece of spectrum has been allocated to some use, there is the additional issue of which carrier is allowed to use which frequencies. Three algorithms were widely used in the past.

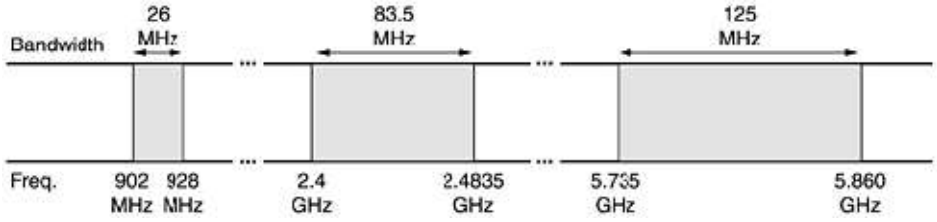
The **oldest algorithm**, often called the **beauty contest**, requires each carrier to explain why its proposal serves the public interest best. Having some government official award property worth billions of dollars to his favorite company often leads to bribery, corruption, nepotism, and worse.

This observation led to **algorithm 2**, holding a lottery among the interested companies. The problem with that idea is that companies with no interest in using the spectrum can enter the lottery. If, say, a fast food restaurant or shoe store chain wins, it can resell the spectrum to a carrier at a huge profit and with no risk.

Algorithm 3: auctioning off the bandwidth to the highest bidder. A completely different approach to allocating frequencies is to not allocate them at all. Just let everyone transmit at will but regulate the power used so that stations have such a short range they do not interfere with each other.

Accordingly, most governments have set aside some frequency bands, called the ISM (Industrial, Scientific, Medical) bands for unlicensed usage. Garage door openers, cordless phones, radio-controlled toys, wireless mice, and numerous other wireless household devices use the ISM bands. To minimize interference between these uncoordinated devices, the FCC mandates that all devices in the ISM bands use spread spectrum techniques. Similar rules apply in other countries. The location of the ISM bands varies somewhat from country to country. In the United States, for example, devices whose power is under 1 watt can use the bands shown in the following figure without requiring a FCC license.

The ISM bands in the United States.



2.2.5 Infrared and Millimeter Waves

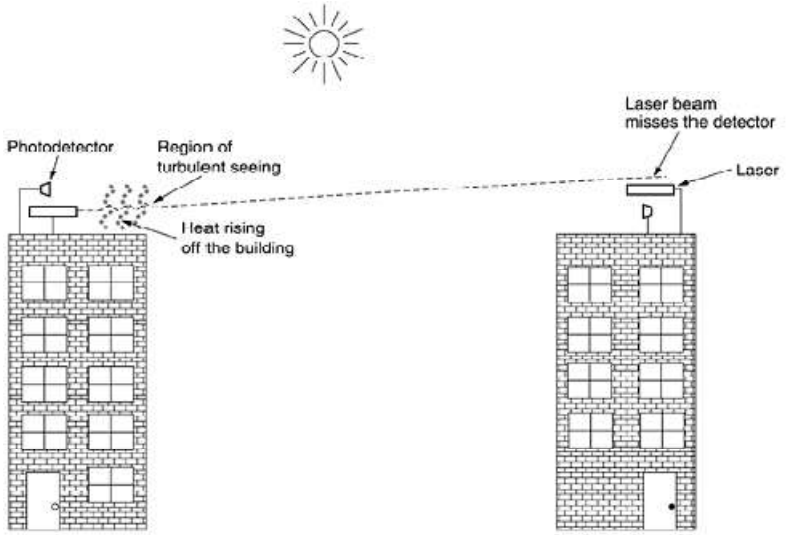
Unguided infrared and millimeter waves are widely used for short-range communication. The remote controls used on televisions, VCRs, and stereos all use infrared communication. They are relatively directional, cheap, and easy to build but have a major drawback: they do not pass through solid objects. In general, as we go from long-wave radio toward visible light, the waves behave more and more like light and less and less like radio.

On the other hand, the fact that infrared waves do not pass through solid walls well is also a plus, means that an infrared system in one room of a building will not interfere with a similar system in adjacent rooms or buildings: you cannot control your neighbor's television with your remote control. No government license is needed to operate an infrared system, in contrast to radio systems, which must be licensed outside the ISM bands. Infrared communication has a limited use on the desktop, for example, connecting notebook computers and printers, but it is not a major player in the communication game.

2.2.6 Lightwave Transmission

Unguided optical signaling has been in use for centuries. A more modern application is to connect the LANs in two buildings via lasers mounted on their rooftops. Coherent optical signaling using lasers is inherently unidirectional, so each building needs its own laser and its own photodetector. This scheme offers very high bandwidth and very low cost. It is also relatively easy to install and, unlike microwave, does not require an FCC license. A disadvantage is that laser beams cannot penetrate rain or thick fog, but they normally work well on sunny days.

Convection currents can interfere with laser communication systems. A bidirectional system with two lasers is pictured here.

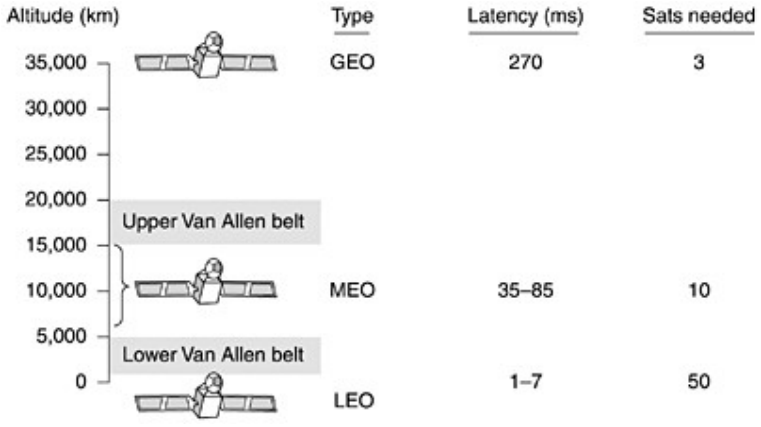


2.3 Communication Satellites

In the 1950s and early 1960s, people tried to set up communication systems by bouncing signals off metalized weather balloons. The received signals were too weak to be of any practical use. Then the U.S. Navy noticed a kind of permanent weather balloon in the sky—the moon—and built an operational system for ship-to-shore communication by bouncing signals off it. Further progress in the celestial communication field had to wait until the first communication satellite was launched. The key difference between an artificial satellite and a real one is that the artificial one can amplify the signals before sending them back, turning a strange curiosity into a powerful communication system.

Communication satellite can be thought of as a big microwave repeater in the sky. It contains several transponders, each of which listens to some portion of the spectrum, amplifies the incoming signal, and then rebroadcasts it at another frequency to avoid interference with the incoming signal. The downward beams can be broad, covering a substantial fraction of the earth's surface, or narrow, covering an area only hundreds of kilometers in diameter. This mode of operation is known as a **bent pipe.** The higher the satellite, the longer the period. It is not the only issue in determining where to place it. Another issue is the presence of the Van Allen belts, layers of highly charged particles trapped by the earth's magnetic field. Any satellite flying within them would be destroyed fairly quickly by the highly-energetic charged particles trapped there by the earth's magnetic field. These factors lead to three regions in which satellites can be placed safely. These regions and some of their properties are illustrated in the following figure.

Communication satellites and some of their properties, including altitude above the earth, round-trip delay time, and number of satellites needed for global coverage.

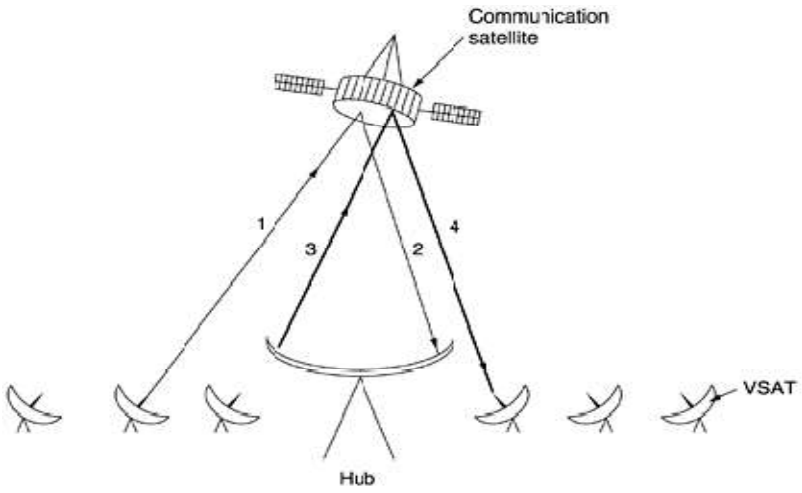


2.3.1 Geostationary Satellites

In 1945, the science fiction writer Arthur C. Clarke calculated that a satellite at an altitude of 35,800 km in a circular equatorial orbit would appear to remain motionless in the sky. so it would not need to be tracked (Clarke, 1945). The first artificial communication satellite, Telstar, was launched in July 1962. These high-flying satellites are often called GEO (Geostationary Earth Orbit) satellites. The effects of solar, lunar, and planetary gravity tend to move them away from their assigned orbit slots and orientations, an effect countered by on-board rocket motors. This fine-tuning activity is called **station keeping**.

However, when the fuel for the motors has been exhausted, typically in about 10 years, the satellite drifts and tumbles helplessly, so it has to be turned off. Eventually, the orbit decays and the satellite reenters the atmosphere and burns up or occasionally crashes to earth. The first geostationary satellites had a single spatial beam that illuminated about 1/3 of the earth's surface, called its **footprint**. A new development in the communication satellite world is the development of low-cost microstations, sometimes called VSATs (Very Small Aperture Terminals). These tiny terminals have 1-meter or smaller antennas and can put out about 1 watt of power. In many VSAT systems, the microstations do not have enough power to communicate directly with one another. Instead, a special ground station, the hub, with a large, high-gain antenna is needed to relay traffic between VSATs, as shown in the following figure.

VSATs using a hub.



2.3.2 Medium-Earth Orbit Satellites

At much lower altitudes, between the two Van Allen belts, we find the MEO (Medium-Earth Orbit) satellites. They have a smaller footprint on the ground and require less powerful transmitters to reach

them. Currently they are not used for telecommunications, the 24 GPS (Global Positioning System) satellites orbiting at about 18,000 km are examples of MEO satellites.

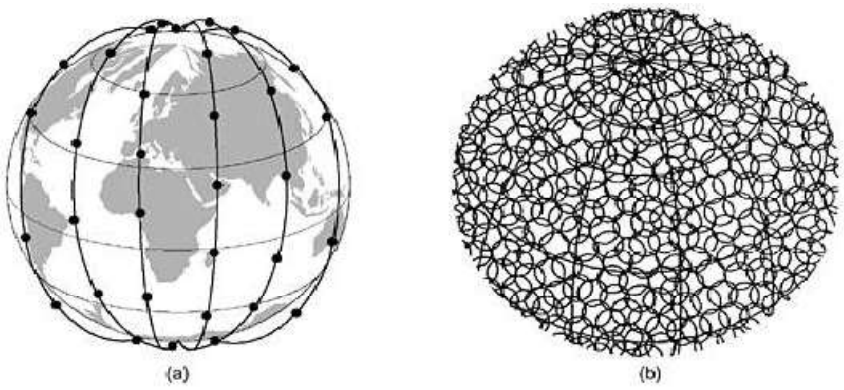
2.3.3 Low-Earth Orbit Satellites

Moving down in altitude, the LEO (Low-Earth Orbit) satellites. Due to their rapid motion, large numbers of them are needed for a complete system. On the other hand, because the satellites are so close to the earth, the ground stations do not need much power, and the round-trip delay is only a few milliseconds. Three examples, two aimed at voice communication and one aimed at Internet service.

2.3.3.1 Iridium

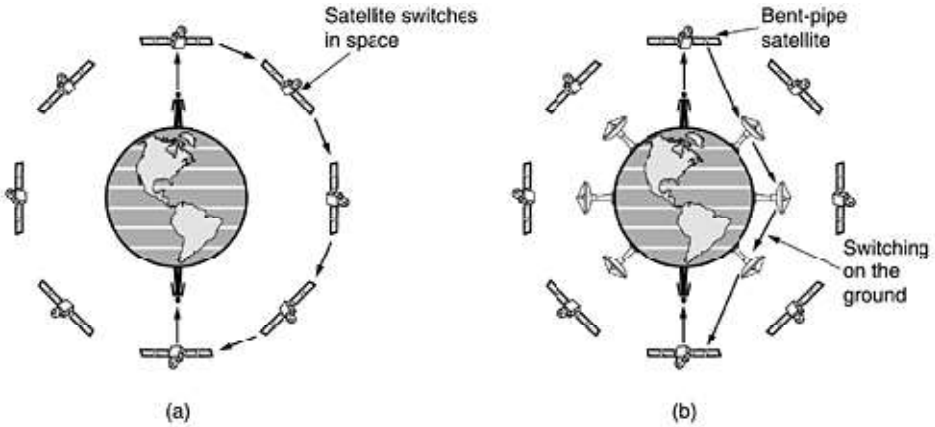
In 1990, Motorola broke new ground by filing an application with the FCC asking for permission to launch 77 low-orbit satellites for the Iridium project (element 77 is iridium). The plan was later revised to use only 66 satellites, so the project should have been renamed Dysprosium (element 66), but that probably sounded too much like a disease. The idea was that as soon as one satellite went out of view, another would replace it. Iridium's business was providing worldwide telecommunication service using hand-held devices that communicate directly with the Iridium satellites. It provides voice, data, paging, fax, and navigation service everywhere on land, sea, and air. The Iridium satellites are positioned at an altitude of 750 km, in circular polar orbits. They are arranged in north south necklaces, with one satellite every 32 degrees of latitude. With six satellite necklaces, the entire earth is covered,

(a) The Iridium satellites form six necklaces around the earth. (b) 1628 moving cells cover the earth.



An interesting property of Iridium is that communication between distant customers takes place in space, with one satellite relaying data to the next one, as illustrated in the following figure. Here we see a caller at the North Pole contacting a satellite directly overhead. The call is relayed via other satellites and finally sent down to the callee at the South Pole.

Relaying in space. (b) Relaying on the ground.



2.3.3.2 Globalstar

An alternative design to Iridium is Globalstar. It is based on 48 LEO satellites but uses a different switching scheme than that of Iridium. Whereas Iridium relays calls from satellite to satellite, which requires sophisticated switching equipment in the satellites. The advantage of this scheme is that it puts much of the complexity on the ground, where it is easier to manage.

2.3.3.3 Teledesic

Teledesic, is targeted at bandwidth-hungry Internet users all over the world. It was conceived in 1990 by mobile phone pioneer Craig McCaw and Microsoft founder Bill Gates. The goal of the Teledesic system is to provide millions of concurrent Internet users with an uplink of as much as 100 Mbps and a downlink of up to 720 Mbps using a small, fixed, VSAT-type antenna, completely bypassing the telephone system. The original design was for a system consisting of 288 small-footprint satellites arranged in 12 planes just below the lower Van Allen belt at an altitude of 1350 km. This was later changed to 30 satellites with larger footprints

2.3.4 Satellites versus Fiber

1. First, while a single fiber has, in principle, more potential bandwidth than all the satellites ever launched, this bandwidth is not available to most users. With satellites, it is practical for a user to erect an antenna on the roof of the building and completely bypass the telephone system to get high bandwidth. Teledesic is based on this idea.
2. Second is for mobile communication, many people now a days want to communicate while jogging, driving, sailing, and flying. Terrestrial fiber optic links are of no use to them, but satellite links potentially are.
3. Third is for situations in which broadcasting is essential. A message sent by satellite can be received by thousands of ground stations at once.
4. Fourth is in market for satellites is to cover areas, where obtaining the right of way for laying fiber is difficult or unduly expensive.
5. Fifth, when rapid deployment is critical, as in military communication systems in time of war, satellites win easily.

6. If advances in technology radically reduce the cost of deploying a satellite (e.g., some future space shuttle can toss out dozens of satellites on one launch) or low-orbit satellites catch on in a big way, it is not certain that fiber will win in all markets.

UNIT-II
The Data Link Layer

3.1 Data Link Layer Design Issues

The data link layer has a number of specific functions it can carry out. These functions include

1. Providing a well-defined service interface to the network layer.
2. Dealing with transmission errors.
3. Regulating the flow of data so that slow receivers are not swamped by fast senders.

To accomplish these goals, the data link layer takes the packets it gets from the network layer and encapsulates them into frames for transmission. Each frame contains a frame header, a payload field for holding the packet, and a frame trailer, as illustrated in the following figure. Frame management forms the heart of what the data link layer does.

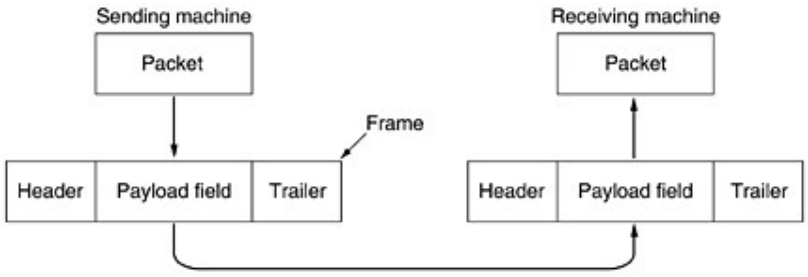


Fig. ***Relationship between packets and frames***

*3.1.1.*Services Provided to the Network Layer

The function of the data link layer is to provide services to the network layer. The principal service is transferring data from the network layer on the source machine to the network layer on the destination machine. On the source machine is an entity, call it a process, in the network layer that hands some bits to the data link layer for transmission to the destination. The job of the data link layer is to transmit the bits to the destination machine so they can be handed over to the network layer there, as shown in the following fig.

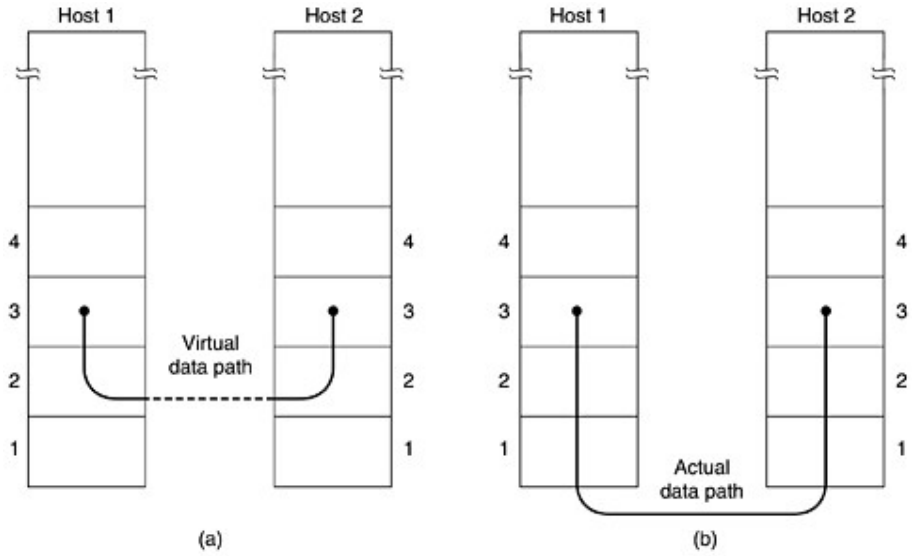


Fig. ***(a) Virtual communication. (b) Actual communication.***

The data link layer can be designed to offer various services. The actual services offered can vary from system to system. Three reasonable possibilities that are commonly provided are

1. **Unacknowledged connectionless service**:

Unacknowledged connectionless service consists of having the source machine send independent frames to the destination machine without having the destination machine acknowledge them. No logical connection is established beforehand or released afterward. If a frame is lost due to noise on the line, no attempt is made to detect the loss or recover from it in the data link layer. This class of service is appropriate when the error rate is very low so that recovery is left to higher layers. It is also appropriate for real-time traffic, such as voice, in which late data are worse than bad data. Most LANs use unacknowledged connectionless service in the data link layer.

2. **Acknowledged connectionless service**:

When this service is offered, there are still no logical connections used, but each frame sent is individually acknowledged. In this way, the sender knows whether a frame has arrived correctly. If it has not arrived within a specified time interval, it can be sent again. This service is useful over unreliable channels, such as wireless systems.

3. **Acknowledged connection-oriented service**:

The most sophisticated service the data link layer can provide to the network layer is connection-oriented service. With this service, the source and destination machines establish a connection before any data are transferred. Each frame sent over the connection is numbered, and the data link layer guarantees that each frame sent is indeed received. Furthermore, it guarantees that each frame is received exactly once and that all frames are received in the right order.

3.1.2 Framing

To provide service to the network layer, the data link layer must use the service provided to it by the physical layer. What the physical layer does is accept a raw bit stream and attempt to deliver it to the destination. This bit stream is not guaranteed to be error free. The number of bits received may be less than, equal to, or more than the number of bits transmitted, and they may have different values. It is up to the data link layer to detect and, if necessary, correct errors.

The usual approach is for the data link layer to break the bit stream up into discrete frames and compute the checksum for each frame. When a frame arrives at the destination, the checksum is recomputed. If the newly-computed checksum is different from the one contained in the frame, the data link layer knows that an error has occurred and takes steps to deal with it (e.g., discarding the bad frame and possibly also sending back an error report).

Breaking the bit stream up into frames is more difficult than it at first appears. One way to achieve this framing is to insert time gaps between frames, much like the spaces between words in ordinary text.

1. Character count.
2. Flag bytes with byte stuffing.
3. Starting and ending flags, with bit stuffing.
4. Physical layer coding violations.

1. **Character count** :

The first framing method uses a field in the header to specify the number of characters in the frame. When the data link layer at the destination sees the character count, it knows how many characters follow and hence where the end of the frame is. This technique is shown in Fig. (a) for four frames of sizes 5, 5, 8, and 8 characters, respectively.

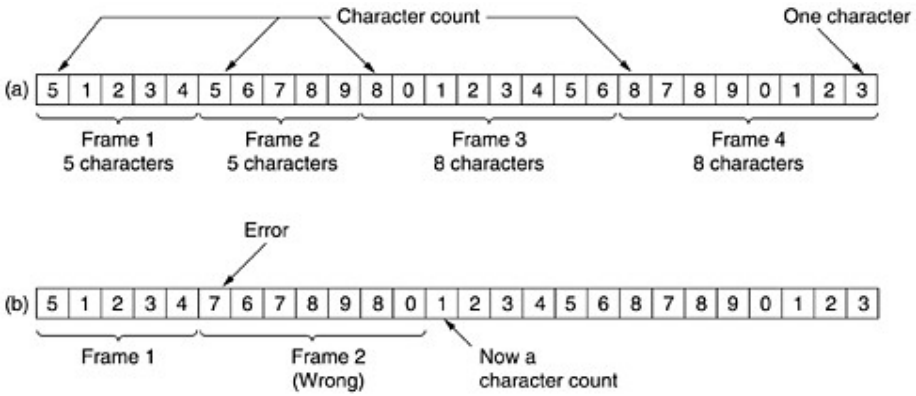


Fig. A character stream. (a) Without errors. (b) With one error.

The trouble with this algorithm is that the count can be garbled by a transmission error.

2. **Flag bytes with byte stuffing**.:

The second framing method gets around the problem of resynchronization after an error by having each frame start and end with special bytes. In the past, the starting and ending bytes were different, but in recent years most protocols have used the same byte, called a flag byte, as both the starting and ending delimiter, as shown in figure as FLAG. In this way, if the receiver ever loses synchronization, it can just search for the flag byte to find the end of the current frame. Two consecutive flag bytes indicate the end of one frame and start of the next one.

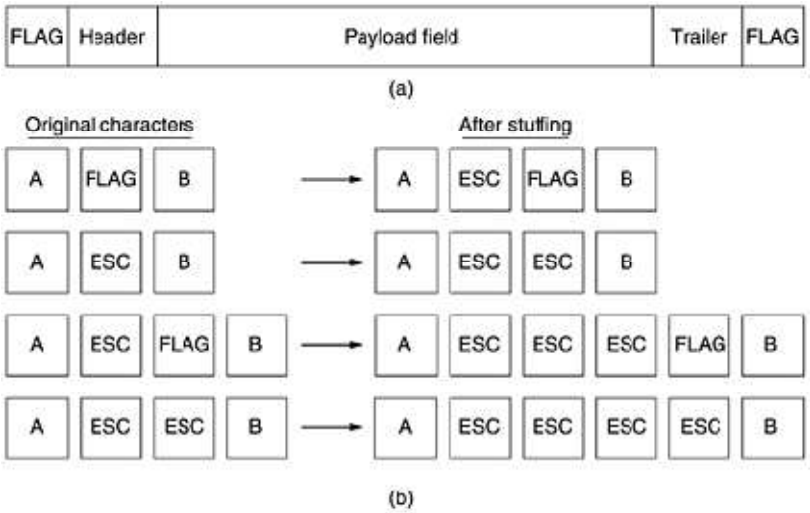


Fig. (a) A frame delimited by flag bytes. (b) Four examples of byte sequences before and after byte stuffing.

A serious problem occurs with this method when binary data, such as object programs or floating-point numbers, are being transmitted. It may easily happen that the flag byte's bit pattern occurs in the data.

This situation will usually interfere with the framing. One way to solve this problem is to have the sender's data link layer insert a special escape byte (ESC) just before each "accidental" flag byte in the data.

The data link layer on the receiving end removes the escape byte before the data are given to the network layer. This technique is called **byte stuffing** or **character stuffing**. Thus, a framing flag byte can be distinguished from one in the data by the absence or presence of an escape byte before it. the next question is: What happens if an escape byte occurs in the middle of the data? The answer is that it, too, is stuffed with an escape byte. Thus, any single escape byte is part of an escape sequence, whereas a doubled one indicates that a single escape occurred naturally in the data.

Some examples are shown in Fig.(b). In all cases, the byte sequence delivered after destuffing is exactly the same as the original byte sequence. A major disadvantage of using this framing method is that it is closely tied to the use of 8-bit characters. Not all character codes use 8-bit characters.

The new technique allows data frames to contain an arbitrary number of bits and allows character codes with an arbitrary number of bits per character. It works like this. Each frame begins and ends with a special bit pattern, 01111110 (in fact, a flag byte). Whenever the sender's data link layer encounters five consecutive 1s in the data, it automatically stuffs a 0 bit into the outgoing bit stream. This bit stuffing is analogous to byte stuffing, in which an escape byte is stuffed into the outgoing character stream before a flag byte in the data.

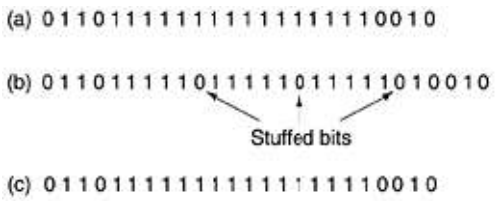


Fig: ***Bit stuffing. (a) The original data. (b) The data as they appear on the line. (c) The data as they are stored in the receiver's memory after destuffing.***

With bit stuffing, the boundary between two frames can be unambiguously recognized by the flag pattern. Thus, if the receiver loses track of where it is, all it has to do is scan the input for flag sequences, since they can only occur at frame boundaries and never within the data

3.1.3 Error Control

Having solved the problem of marking the start and end of each frame, come to the next problem: how to make sure all frames are eventually delivered to the network layer at the destination and in the proper order. Suppose that the sender just kept outputting frames without regard to whether they were arriving

properly. This might be fine for unacknowledged connectionless service, but would most certainly not be fine for reliable, connection-oriented service.

The usual way to ensure reliable delivery is to provide the sender with some feedback about what is happening at the other end of the line. Typically, the protocol calls for the receiver to send back special control frames bearing positive or negative acknowledgements about the incoming frames. If the sender receives a positive acknowledgement about a frame, it knows the frame has arrived safely. On the other hand, a negative acknowledgement means that something has gone wrong, and the frame must be transmitted again.

However, if either the frame or the acknowledgement is lost, the timer will go off, alerting the sender to a potential problem. The obvious solution is to just transmit the frame again. However, when frames may be transmitted multiple times there is a danger that the receiver will accept the same frame two or more times and pass it to the network layer more than once. To prevent this from happening, it is generally necessary to assign sequence numbers to outgoing frames, so that the receiver can distinguish retransmissions from originals.

3.1.4.Flow Control

Another important design issue that occurs in the data link layer (and higher layers as well) is what to do with a sender that systematically wants to transmit frames faster than the receiver can accept them. This situation can easily occur when the sender is running on a fast (or lightly loaded) computer and the receiver is running on a slow (or heavily loaded) machine. The sender keeps pumping the frames out at a high rate until the receiver is completely swamped. Even if the transmission is error free, at a certain point the receiver will simply be unable to handle the frames as they arrive and will start to lose some. Clearly, something has to be done to prevent this situation.

Two approaches are commonly used. In the first one, **feedback-based flow control**, the receiver sends back information to the sender giving it permission to send more data or at least telling the sender how the receiver is doing. In the second one, **rate-based flow control**, the protocol has a built-in mechanism that limits the rate at which senders may transmit data, without using feedback from the receiver

3.2.Error Detection and Correction

The telephone system has three parts: **the switches**, the **interoffice trunks**, and the **local loops**. The first two are now almost entirely digital in most developed countries. The local loops are still analog twisted copper pairs and will continue to be so for years due to the enormous expense of replacing them. While errors are rare on the digital part, they are still common on the local loops. Furthermore, wireless communication is becoming more common, and the error rates here are orders of magnitude worse than on the interoffice fiber trunks.

As a result of the physical processes that generate them, errors on some media (e.g., radio) tend to come in bursts rather than singly. Having the errors come in bursts has both advantages and disadvantages over isolated single-bit errors. On the advantage side, computer data are always sent in blocks of bits. Suppose that the block size is 1000 bits and the error rate is 0.001 per bit. If errors were independent, most blocks would contain an error. If the errors came in bursts of 100 however, only one or two blocks in 100 would be affected, on average. The disadvantage of burst errors is that they are much harder to correct than are isolated errors.

3.2.1 Error-Correcting Codes

Network designers have developed two basic strategies for dealing with errors. One way is to include enough redundant information along with each block of data sent, to enable the receiver to deduce what the transmitted data must have been. The other way is to include only enough redundancy to allow the receiver to deduce that an error occurred, but not which error, and have it request a retransmission. The former strategy uses error correcting codes and the latter uses error-detecting codes. The use of error-correcting codes is often referred to as **forward error correction**.

Each of these techniques occupies a different ecological niche. On channels that are highly reliable, such as fiber, it is cheaper to use an error detecting code and just retransmit the occasional block found to be faulty. However, on channels such as wireless links that make many errors, it is better to add enough redundancy to each block for the receiver to be able to figure out what the original block was, rather than relying on a retransmission, which itself may be in error.

To understand how errors can be handled, it is necessary to look closely at what an error really is. Normally, a frame consists of m data (i.e., message) bits and r redundant, or check, bits. Let the total length be n (i.e., n = m + r). An n-bit unit containing data and check bits is often referred to as an n-bit codeword

Given any two codewords, say, 10001001 and 10110001, it is possible to determine how many corresponding bits differ. In this case, 3 bits differ. To determine how many bits differ, just exclusive OR the two codewords and count the number of 1 bits in the result, for example:

The number of bit positions in which two codewords differ is called the Hamming distance (Hamming, 1950). Its significance is that if two codewords are a Hamming distance d apart, it will require d single-bit errors to convert one into the other.

Hamming Code

Hamming code is a block code that is capable of detecting up to two simultaneous bit errors and correcting single-bit errors. It was developed by R.W. Hamming for error correction.

In this coding method, the source encodes the message by inserting redundant bits within the message. These redundant bits are extra bits that are generated and inserted at specific positions in the message itself to enable error detection and correction. When the destination receives this message, it performs recalculations to detect errors and find the bit position that has error.

Encoding a message by Hamming Code

The procedure used by the sender to encode the message encompasses the following steps −

- **Step 1** − Calculation of the number of redundant bits.
- **Step 2** − Positioning the redundant bits.
- **Step 3** − Calculating the values of each redundant bit.

Once the redundant bits are embedded within the message, this is sent to the user.

Step 1 − Calculation of the number of redundant bits.

If the message contains *mm*number of data bits, *rr*number of redundant bits are added to it so that *mr* is able to indicate at least ($m + r$+ 1) different states. Here, ($m + r$) indicates location of an error in each of ($m + r$) bit positions and one additional state indicates no error. Since, *rr* bits can indicate $2^r r$ states, $2^r r$ must be at least equal to ($m + r + 1$). Thus the following equation should hold $2^r \geq m+r+1$

Step 2 − Positioning the redundant bits.

The *r* redundant bits placed at bit positions of powers of 2, i.e. 1, 2, 4, 8, 16 etc. They are referred in the rest of this text as r_1 (at position 1), r_2 (at position 2), r_3 (at position 4), r_4 (at position 8) and so on.

Step 3 − Calculating the values of each redundant bit.

The redundant bits are parity bits. A parity bit is an extra bit that makes the number of 1s either even or odd. The two types of parity are −

- **Even Parity** − Here the total number of bits in the message is made even.
- **Odd Parity** − Here the total number of bits in the message is made odd.

Each redundant bit, r_i, is calculated as the parity, generally even parity, based upon its bit position. It covers all bit positions whose binary representation includes a 1 in the i^{th} position except the position of r_i. Thus −

- r_1 is the parity bit for all data bits in positions whose binary representation includes a 1 in the least significant position excluding 1 (3, 5, 7, 9, 11 and so on)
- r_2 is the parity bit for all data bits in positions whose binary representation includes a 1 in the position 2 from right except 2 (3, 6, 7, 10, 11 and so on)
- r_3 is the parity bit for all data bits in positions whose binary representation includes a 1 in the position 3 from right except 4 (5-7, 12-15, 20-23 and so on)

Decoding a message in Hamming Code

Once the receiver gets an incoming message, it performs recalculations to detect errors and correct them. The steps for recalculation are −

- **Step 1** − Calculation of the number of redundant bits.
- **Step 2** − Positioning the redundant bits.
- **Step 3** − Parity checking.
- **Step 4** − Error detection and correction

Step 1 − Calculation of the number of redundant bits

Using the same formula as in encoding, the number of redundant bits are ascertained.

$2^r \geq m + r + 1$ where *m* is the number of data bits and *r* is the number of redundant bits.

Step 2 − Positioning the redundant bits

The *r* redundant bits placed at bit positions of powers of 2, i.e. 1, 2, 4, 8, 16 etc.

Step 3 − Parity checking

Parity bits are calculated based upon the data bits and the redundant bits using the same rule as during generation of c_1,c_2 ,c_3 ,c_4 etc. Thus

c_1 = parity(1, 3, 5, 7, 9, 11 and so on)

c_2 = parity(2, 3, 6, 7, 10, 11 and so on)

c_3 = parity(4-7, 12-15, 20-23 and so on)

Step 4 − Error detection and correction

The decimal equivalent of the parity bits binary values is calculated. If it is 0, there is no error. Otherwise, the decimal value gives the bit position which has error. For example, if $c_1c_2c_3c_4$ = 1001, it implies that the data bit at position 9, decimal equivalent of 1001, has error. The bit is flipped to get the correct message.

Error-correcting codes are widely used on wireless links, which are notoriously noisy and error prone when compared to copper wire or optical fibers. Without error-correcting codes, it would be hard to get anything through. However, over copper wire or fiber, the error rate is much lower, so error detection and retransmission is usually more efficient there for dealing with the occasional error.

The algorithm for computing the checksum is as follows:
1. Let r be the degree of G(x). Append r zero bits to the low-order end of the frame so it now contains m + r bits and corresponds to the polynomial xrM(x).
2. Divide the bit string corresponding to G(x) into the bit string corresponding to xrM(x), using modulo 2 division.
3. Subtract the remainder (which is always r or fewer bits) from the bit string corresponding to xrM(x) using modulo 2 subtraction. The result is the checksummed frame to be transmitted. Call its polynomial T(x). The following figure illustrates the calculation for a frame 1101011011 using the generator G(x) = x4 + x + 1.

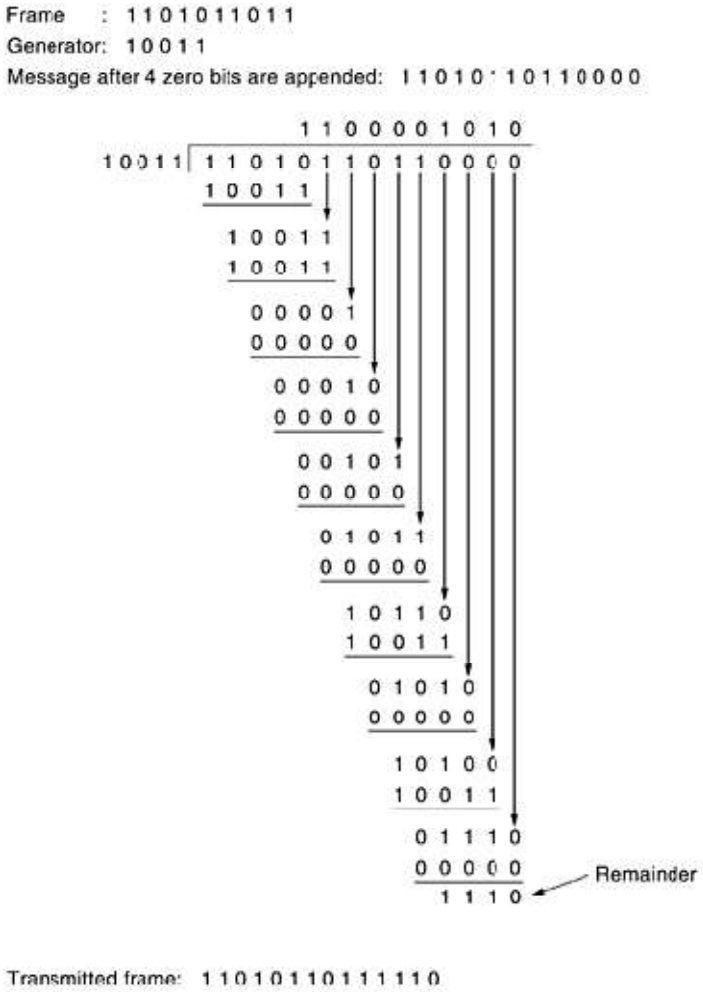


Fig. Calculation of the polynomial code checksum

3.3 Elementary Data Link Protocols

3.3.1 An Unrestricted Simplex Protocol

Data are transmitted in one direction only. Both the transmitting and receiving network layers are always ready. Processing time can be ignored. Infinite buffer space is available. And best of all, the communication channel between the data link layers never damages or loses frames. This thoroughly unrealistic protocol, which we will nickname "utopia," is shown in Fig.

```
/* Protocol 1 (utopia) provides for data transmission in one direction only, from
   sender to receiver.  The communication channel is assumed to be error free
   and the receiver is assumed to be able to process all the input infinitely quickly.
   Consequently, the sender just sits in a loop pumping data out onto the line as
   fast as it can. */

typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender1(void)
{
  frame s;                          /* buffer for an outbound frame */
  packet buffer;                    /* buffer for an outbound packet */

  while (true) {
      from_network_layer(&buffer);  /* go get something to send */
      s.info = buffer;              /* copy it into s for transmission */
      to_physical_layer(&s);        /* send it on its way */
  }                                 /* Tomorrow, and tomorrow, and tomorrow,
                                       Creeps in this petty pace from day to day
                                       To the last syllable of recorded time.
                                           - Macbeth, V, v */
}


void receiver1(void)
{
  frame r;
  event_type event;                 /* filled in by wait, but not used here */

  while (true) {
      wait_for_event(&event);       /* only possibility is frame_arrival */
      from_physical_layer(&r);      /* go get the inbound frame */
      to_network_layer(&r.info);    /* pass the data to the network layer */
  }
}
```

Fig. An unrestricted simplex protocol.

The protocol consists of two distinct procedures, a sender and a receiver. The sender runs in the data link layer of the source machine, and the receiver runs in the data link layer of the destination machine. No sequence numbers or acknowledgements are used here, so MAX_SEQ is not needed. The only event type possible is frame_arrival

The sender is in an infinite while loop just pumping data out onto the line as fast as it can. The body of the loop consists of three actions: go fetch a packet from the (always obliging) network layer, construct an outbound frame using the variable s, and send the frame on its way. Only the info field of the frame is used

by this protocol, because the other fields have to do with error and flow control and there are no errors or flow control restrictions here.

The receiver is equally simple. Initially, it waits for something to happen, the only possibility being the arrival of an undamaged frame. Eventually, the frame arrives and the procedure wait_for_event returns, with event set_to_frame_arrival (which is ignored anyway). The call to from_physical_layer removes the newly arrived frame from the hardware.

3.3.2 A Simplex Stop-and-Wait Protocol

The main problem we have to deal with here is how to prevent the sender from flooding the receiver with data faster than the latter is able to process them. In essence, if the receiver requires a time _t to execute from_physical_layer plus to_network_layer, the sender must transmit at an average rate less than one frame per time _t. Moreover, if we assume that no automatic buffering and queueing are done within the receiver's hardware, the sender must never transmit a new frame until the old one has been fetched by from_physical_layer, lest the new one overwrite the old one.

A more general solution to this dilemma is to have the receiver provide feedback to the sender. After having passed a packet to its network layer, the receiver sends a little dummy frame back to the sender which, in effect, gives the sender permission to transmit the next frame. After having sent a frame, the sender is required by the protocol to bide its time until the little dummy (i.e., acknowledgement) frame arrives. Using feedback from the receiver to let the sender know when it may send more data is an example of the flow control mentioned earlier. Protocols in which the sender sends one frame and then waits for an acknowledgement before proceeding are called stop-and-wait. Following figure gives an example of a simplex stop-and-wait protocol.

Although data traffic in this example is simplex, going only from the sender to the receiver, frames do travel in both directions. Consequently, the communication channel between the two data link layers needs to be capable of bidirectional information transfer. However, this protocol entails a strict alternation of flow: first the sender sends a frame, then the receiver sends a frame, then the sender sends another frame, then the receiver sends another one, and so on. A half- duplex physical channel would suffice here.

```
/* Protocol 2 (stop-and-wait) also provides for a one-directional flow of data from
   sender to receiver. The communication channel is once again assumed to be error
   free, as in protocol 1. However, this time, the receiver has only a finite buffer
   capacity and a finite processing speed, so the protocol must explicitly prevent
   the sender from flooding the receiver with data faster than it can be handled. */

typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender2(void)
{
  frame s;                          /* buffer for an outbound frame */
  packet buffer;                    /* buffer for an outbound packet */
  event_type event;                 /* frame_arrival is the only possibility */

  while (true) {
      from_network_layer(&buffer);  /* go get something to send */
      s.info = buffer;              /* copy it into s for transmission */
      to_physical_layer(&s);        /* bye-bye little frame */
      wait_for_event(&event);       /* do not proceed until given the go ahead */
  }
}

void receiver2(void)
{
  frame r, s;                       /* buffers for frames */
  event_type event;                 /* frame_arrival is the only possibility */
  while (true) {
      wait_for_event(&event);       /* only possibility is frame_arrival */
      from_physical_layer(&r);      /* go get the inbound frame */
      to_network_layer(&r.info);    /* pass the data to the network layer */
      to_physical_layer(&s);        /* send a dummy frame to awaken sender */
  }
}
```

Fig. ***A simplex stop-and-wait protocol***

3.3.3 A Simplex Protocol for a Noisy Channel

Now let us consider the normal situation of a communication channel that makes errors. Frames may be either damaged or lost completely. However, we assume that if a frame is damaged in transit, the receiver hardware will detect this when it computes the checksum. If the frame is damaged in such a way that the checksum is nevertheless correct, an unlikely occurrence, this protocol (and all other protocols) can fail (i.e., deliver an incorrect packet to the network layer).

To see what might go wrong, remember that it is the task of the data link layer processes to provide error-free, transparent communication between network layer processes. The network layer on machine A gives a series of packets to its data link layer, which must ensure that an identical series of packets are delivered to the network layer on machine B by its data link layer. In particular, the network layer on B has

no way of knowing that a packet has been lost or duplicated, so the data link layer must guarantee that no combination of transmission errors, however unlikely, can cause a duplicate packet to be delivered to a network layer. Consider the following scenario:

1. The network layer on A gives packet 1 to its data link layer. The packet is correctly received at B and passed to the network layer on B. B sends an acknowledgement frame back to A.
2. The acknowledgement frame gets lost completely. It just never arrives at all. Life would be a great deal simpler if the channel mangled and lost only data frames and not control frames, but sad to say, the channel is not very discriminating.
3. The data link layer on A eventually times out. Not having received an acknowledgement, it (incorrectly) assumes that its data frame was lost or damaged and sends the frame containing packet 1 again.
4. The duplicate frame also arrives at the data link layer on B perfectly and is unwittingly passed to the network layer there. If A is sending a file to B, part of the file will be duplicated (i.e., the copy of the file made by B will be incorrect and the error will not have been detected). In other words, the protocol will fail.

Protocols in which the sender waits for a positive acknowledgement before advancing to the next data item are often called PAR (Positive Acknowledgement with Retransmission) or ARQ (Automatic Repeat reQuest)

```
/* Protocol 3 (par) allows unidirectional data flow over an unreliable channel. */
#define MAX_SEQ 1                                   /* must be 1 for protocol 3 */
typedef enum {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"

void sender3(void)
{
  seq_nr next_frame_to_send;                        /* seq number of next outgoing frame */
  frame s;                                          /* scratch variable */
  packet buffer;                                    /* buffer for an outbound packet */
  event_type event;

  next_frame_to_send = 0;                           /* initialize outbound sequence numbers */
  from_network_layer(&buffer);                      /* fetch first packet */
  while (true) {
      s.info = buffer;                              /* construct a frame for transmission */
      s.seq = next_frame_to_send                    /* insert sequence number in frame */
      to_physical_layer(&s);                        /* send it on its way */
      start_timer(s.seq);                           /* if answer takes too long, time out */
      wait_for_event(&event);                       /* frame_arrival, cksum_err, timeout */
      if (event == frame_arrival) {
            from_physical_layer(&s);                /* get the acknowledgement */
            if (s.ack == next_frame_to_send) {
                  stop_timer(s.ack);                /* turn the timer off */
                  from_network_layer(&buffer);      /* get the next one to send */
                  inc(next_frame_to_send);          /* invert next_frame_to_send */
            }
      }
  }
}

void receiver3(void)
{
  seq_nr frame_expected;
  frame r, s;
  event_type event;

  frame_expected = 0;
  while (true) {
      wait_for_event(&event);                       /* possibilities: frame_arrival, cksum_err */
      if (event == frame_arrival) {                 /* a valid frame has arrived. */
            from_physical_layer(&r);                /* go get the newly arrived frame */
            if (r.seq == frame_expected) {          /* this is what we have been waiting for. */
                  to_network_layer(&r.info);        /* pass the data to the network layer */
                  inc(frame_expected);              /* next time expect the other sequence nr */
            }
            s.ack = 1 - frame_expected;             /* tell which frame is being acked */
            to_physical_layer(&s);                  /* send acknowledgement */
      }
  }
}
```

Fig. A positive acknowledgement with retransmission protocol

After transmitting a frame and starting the timer, the sender waits for something exciting to happen. Only three possibilities exist: an acknowledgement frame arrives undamaged, a damaged acknowledgement frame staggers in, or the timer expires. If a valid acknowledgement comes in, the sender fetches the next packet from its network layer and puts it in the buffer, overwriting the previous packet. It also advances the sequence number. If a damaged frame arrives or no frame at all arrives, neither the buffer nor the sequence number is changed so that a duplicate can be sent.

When a valid frame arrives at the receiver, its sequence number is checked to see if it is a duplicate. If not, it is accepted, passed to the network layer, and an acknowledgement is generated. Duplicates and damaged frames are not passed to the network layer.

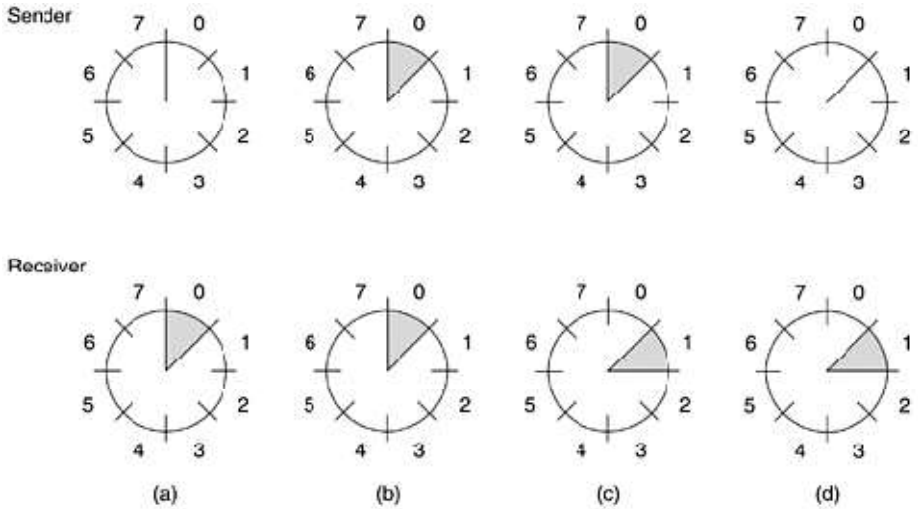
3.4 Sliding Window Protocols

In the previous protocols, data frames were transmitted in one direction only. In most practical situations, there is a need for transmitting data in both directions. One way of achieving full-duplex data transmission is to have two separate communication channels and use each one for simplex data traffic (in different directions). If this is done, we have two separate physical circuits, each with a "forward" channel (for data) and a "reverse" channel (for acknowledgements). In both cases the bandwidth of the reverse channel is almost entirely wasted. In effect, the user is paying for two circuits but using only the capacity of one.

The essence of all sliding window protocols is that at any instant of time, the sender maintains a set of sequence numbers corresponding to frames it is permitted to send. These frames are said to fall within the sending window. Similarly, the receiver also maintains a receiving window corresponding to the set of frames it is permitted to accept. The sender's window and the receiver's window need not have the same lower and upper limits or even have the same size. In some protocols they are fixed in size, but in others they can grow or shrink over the course of time as frames are sent and received.

Although these protocols give the data link layer more freedom about the order in which it may send and receive frames, we have definitely not dropped the requirement that the protocol must deliver packets to the destination network layer in the same order they were passed to the data link layer on the sending machine. Nor have we changed the requirement that the physical communication channel is "wire-like," that is, it must deliver all frames in the order sent.

The sequence numbers within the sender's window represent frames that have been sent or can be sent but are as yet not acknowledged. Whenever a new packet arrives from the network layer, it is given the next highest sequence number, and the upper edge of the window is advanced by one. When an acknowledgement comes in, the lower edge is advanced by one. In this way the window continuously maintains a list of unacknowledged frames. Following figure shows an example.



A sliding window of size 1, with a 3-bit sequence number. (a) Initially. (b) After the first frame has been sent. (c) After the first frame has been received. (d) After the first acknowledgement has been received.

```
/* Protocol 4 (sliding window) is bidirectional. */

#define MAX_SEQ 1                       /* must be 1 for protocol 4 */
typedef enum {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"
void protocol4 (void)
{
  seq_nr next_frame_to_send;            /* 0 or 1 only */
  seq_nr frame_expected;                /* 0 or 1 only */
  frame r, s;                           /* scratch variables */
  packet buffer;                        /* current packet being sent */
  event_type event;
  next_frame_to_send = 0;               /* next frame on the outbound stream */
  frame_expected = 0;                   /* frame expected next */
  from_network_layer(&buffer);          /* fetch a packet from the network layer */
  s.info = buffer;                      /* prepare to send the initial frame */
  s.seq = next_frame_to_send;           /* insert sequence number into frame */
  s.ack = 1 - frame_expected;           /* piggybacked ack */
  to_physical_layer(&s);                /* transmit the frame */
  start_timer(s.seq);                   /* start the timer running */
  while (true) {
      wait_for_event(&event);           /* frame_arrival, cksum_err, or timeout */
      if (event == frame_arrival) {     /* a frame has arrived undamaged. */
          from_physical_layer(&r);      /* go get it */
          if (r.seq == frame_expected) {/* handle inbound frame stream. */
              to_network_layer(&r.info);/* pass packet to network layer */
              inc(frame_expected);      /* invert seq number expected next */
          }
          if (r.ack == next_frame_to_send) {/* handle outbound frame stream. */
              stop_timer(r.ack);        /* turn the timer off */
              from_network_layer(&buffer);/* fetch new pkt from network layer */
              inc(next_frame_to_send);  /* invert sender's sequence number */
          }
      }
      s.info = buffer;                  /* construct outbound frame */
      s.seq = next_frame_to_send;       /* insert sequence number into it */
      s.ack = 1 - frame_expected;       /* seq number of last received frame */
      to_physical_layer(&s);            /* transmit a frame */
      start_timer(s.seq);               /* start the timer running */
  }
}
```

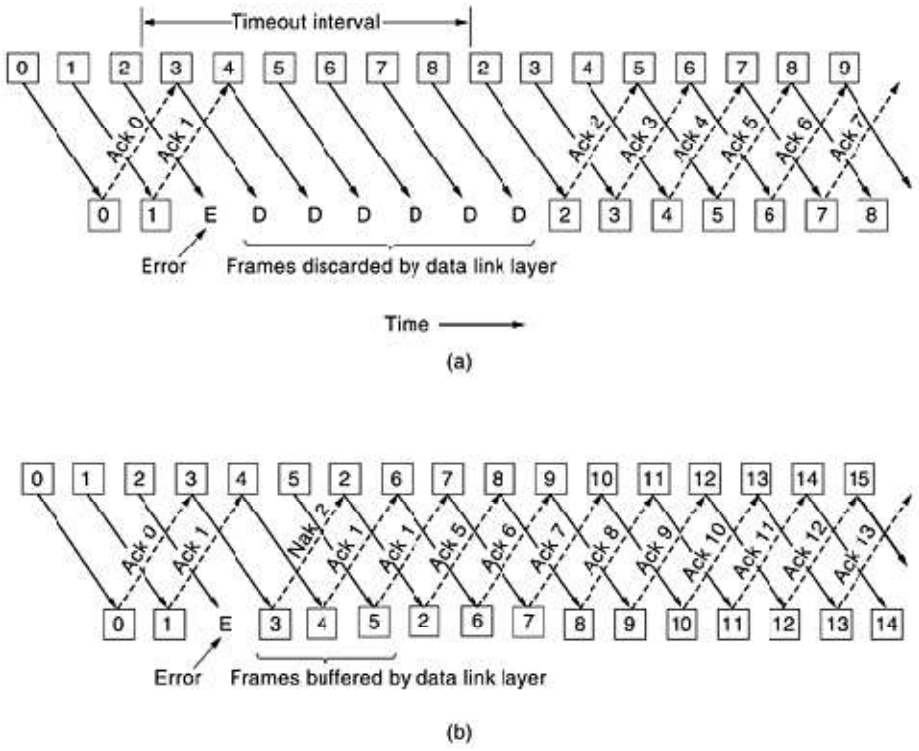
A 1-bit sliding window protocol

3.4.2 A Protocol Using Go Back N

Until now we have made the tacit assumption that the transmission time required for a frame to arrive at the receiver plus the transmission time for the acknowledgement to come back is negligible. Sometimes this assumption is clearly false. In these situations the long round-trip time can have important implications for the efficiency of the bandwidth utilization. The problem described above can be viewed as a consequence of the rule requiring a sender to wait for an acknowledgement before sending another frame. If we relax that restriction, much better efficiency can be achieved. Basically, the solution lies in allowing the sender to transmit up to w frames before blocking, instead of just 1. With an appropriate choice of w the sender will be able to continuously transmit frames for a time equal to the round-trip transit time without filling up the window.

The need for a large window on the sending side occurs whenever the product of bandwidth x round-trip-delay is large. If the bandwidth is high, even for a moderate delay, the sender will exhaust its window quickly unless it has a large window. If the delay is high (e.g., on a geostationary satellite channel), the sender will exhaust its window even for a moderate bandwidth. The product of these two factors basically tells what the capacity of the pipe is, and the sender needs the ability to fill it without stopping in order to operate at peak efficiency.

This technique is known as pipelining Pipelining frames over an unreliable communication channel raises some serious issues



Pipelining and error recovery. Effect of an error when (a) receiver's window size is 1 and (b) receiver's window size is large.

Two basic approaches are available for dealing with errors in the presence of pipelining. One way, called go back n, is for the receiver simply to discard all subsequent frames, sending no acknowledgements for the discarded frames. This strategy corresponds to a receive window of size 1. In other words, the data link layer refuses to accept any frame except the next one it must give to the network layer. If the sender's

window fills up before the timer runs out, the pipeline will begin to empty. Eventually, the sender will time out and retransmit all unacknowledged frames in order, starting with the damaged or lost one. This approach can waste a lot of bandwidth if the error rate is high.

```
/* Protocol 5 (go back n) allows multiple outstanding frames. The sender may transmit up
   to MAX_SEQ frames without waiting for an ack. In addition, unlike in the previous
   protocols, the network layer is not assumed to have a new packet all the time. Instead,
   the network layer causes a network_layer_ready event when there is a packet to send. */

#define MAX_SEQ 7                       /* should be 2^n - 1 */
typedef enum {frame_arrival, cksum_err, timeout, network_layer_ready} event_type;
#include "protocol.h"

static boolean between(seq_nr a, seq_nr b, seq_nr c)
{
/* Return true if a <=b < c circularly; false otherwise. */
  if (((a <= b) && (b < c)) || ((c < a) && (a <= b)) || ((b < c) && (c < a)))
      return(true);
    else
      return(false);
}

static void send_data(seq_nr frame_nr, seq_nr frame_expected, packet buffer[ ])
{
/* Construct and send a data frame. */
  frame s;                              /* scratch variable */

  s.info = buffer[frame_nr];            /* insert packet into frame */
  s.seq = frame_nr;                     /* insert sequence number into frame */
  s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1);/* piggyback ack */
  to_physical_layer(&s);                /* transmit the frame */
  start_timer(frame_nr);                /* start the timer running */
}

void protocol5(void)
{
  seq_nr next_frame_to_send;            /* MAX_SEQ > 1; used for outbound stream */
  seq_nr ack_expected;                  /* oldest frame as yet unacknowledged */
  seq_nr frame_expected;                /* next frame expected on inbound stream */
  frame r;                              /* scratch variable */
  packet buffer[MAX_SEQ + 1];           /* buffers for the outbound stream */
  seq_nr nbuffered;                     /* # output buffers currently in use */
  seq_nr i;                             /* used to index into the buffer array */
  event_type event;

  enable_network_layer();               /* allow network_layer_ready events */
  ack_expected = 0;                     /* next ack expected inbound */
  next_frame_to_send = 0;               /* next frame going out */
  frame_expected = 0;                   /* number of frame expected inbound */
  nbuffered = 0;                        /* initially no packets are buffered */

  while (true) {
    wait_for_event(&event);             /* four possibilities: see event_type above */

    switch(event) {
      case network_layer_ready:         /* the network layer has a packet to send */
           /* Accept, save, and transmit a new frame. */
           from_network_layer(&buffer[next_frame_to_send]); /* fetch new packet */
           nbuffered = nbuffered + 1;   /* expand the sender's window */
           send_data(next_frame_to_send, frame_expected, buffer);/* transmit the frame */
           inc(next_frame_to_send);     /* advance sender's upper window edge */
           break;

      case frame_arrival:               /* a data or control frame has arrived */
           from_physical_layer(&r);     /* get incoming frame from physical layer */

           if (r.seq == frame_expected) {
                /* Frames are accepted only in order. */
                to_network_layer(&r.info);  /* pass packet to network layer */
                inc(frame_expected);        /* advance lower edge of receiver's window */
           }
```

3.4.3 A Protocol Using Selective Repeat

Protocol 5 works well if errors are rare, but if the line is poor, it wastes a lot of bandwidth on retransmitted frames. An alternative strategy for handling errors is to allow the receiver to accept and buffer the frames following a damaged or lost one. Such a protocol does not discard frames merely because an earlier frame was damaged or lost. In this protocol, both sender and receiver maintain a window of acceptable sequence numbers. The sender's window size starts out at 0 and grows to some predefined maximum, MAX_SEQ. The receiver's window, in contrast, is always fixed in size and equal to MAX_SEQ. The receiver has a buffer reserved for each sequence number within its fixed window. Associated with each buffer is a bit (arrived) telling whether the buffer is full or empty.

Whenever a frame arrives, its sequence number is checked by the function between to see if it falls within the window. If so and if it has not already been received, it is accepted and stored. This action is taken without regard to whether or not it contains the next packet expected by the network layer. Of course, it must be kept within the data link layer and not passed to the network layer until all the lower-numbered frames have already been delivered to the network layer in the correct order. A protocol using this algorithm is given in following fig.

The Medium Access Control Sublayer

Networks can be divided into two categories: those using point-to- point connections and those using broadcast channels. In any broadcast network, the key issue is how to determine who gets to use the channel? when there is competition for it?. When only a single channel is available, determining who should go next is much harder. Many protocols. In the literature, broadcast channels are sometimes referred to as **multi access channels** or **random access channels**. The protocols used to determine who goes next on a multi access channel belong to a sublayer of the data link layer called the **MAC** (**Medium Access Control**) sublayer. Technically, the MAC sublayer is the bottom part of the data link layer

1. Multiple Access Protocols

1.1 ALOHA

In the 1970s, Norman Abramson and his colleagues at the University of Hawaii devised a new and elegant method to solve the channel allocation problem. ALOHA system, used ground-based radio

broadcasting, the basic idea is applicable to any system in which uncoordinated users are competing for the use of a single shared channel. There are two versions of ALOHA here: pure and slotted.

1.1.1Pure ALOHA

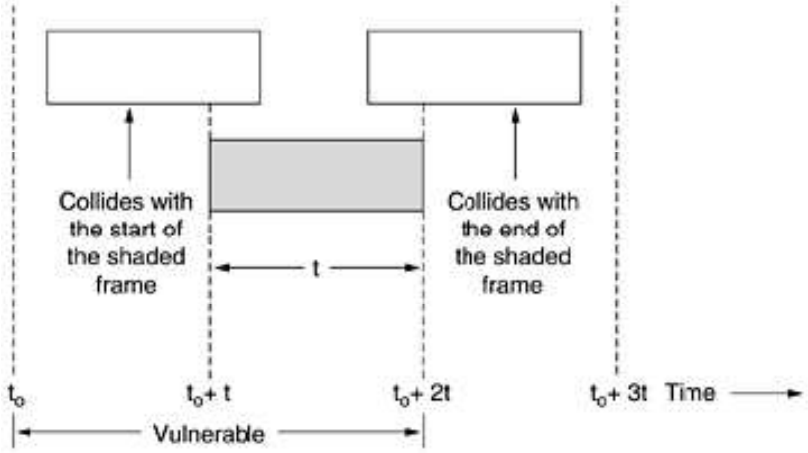
The basic idea of an ALOHA system is simple: let users transmit whenever they have data to be sent. There will be collisions, of course, and the colliding frames will be damaged. With a LAN, the feedback is immediate; with a satellite, there is a delay of 270 msec before the sender knows if the transmission was successful. If the frame was destroyed, the sender just waits a random amount of time and sends it again. The waiting time must be random or the same frames will collide over and over, in lockstep. Systems in which multiple users share a common channel in a way that can lead to conflicts are widely known as **contention** systems. All frames are with the same length because the throughput of ALOHA systems is maximized by having a uniform frame size rather than by allowing variable length frames.

Whenever two frames try to occupy the channel at the same time, there will be a collision and both will be garbled. If the first bit of a new frame overlaps with just the last bit of a frame almost finished, both frames will be totally destroyed and both will have to be retransmitted later. The checksum cannot (and should not) distinguish between a total loss and a near miss.

What is the efficiency of an ALOHA channel?

Let the "frame time" denote the amount of time needed to transmit the standard, fixed-length Frame At high load there will be many collisions. Under all loads, the throughput, S, is just the offered load, G, times the probability, $P0$, of a transmission succeeding—that is, $S = GP0$, where $P0$ is the probability that a frame does not suffer a collision. A frame will not suffer a collision if no other frames are sent within one frame time of its start, as shown in following figure. Under what conditions will the shaded frame arrive undamaged? Let t be the time required to send a frame. If any other user has generated a frame between time $t0$ and $t0 + t$, the end of that frame will collide with the beginning of the shaded one. In pure ALOHA a station does not listen to the channel before transmitting, it has no way of knowing that another frame was already underway. Similarly, any other frame started between $t0 + t$ and $t0 + 2t$ will bump into the end of the shaded frame.

Vulnerable period for the shaded frame.



So the probability of zero frames is just *e*-*G*. In an interval two frame times long, the mean number of frames generated is 2*G*. The probability of no other traffic being initiated during the entire vulnerable period is thus given by $P0 = e^{-2G}$. Using $S = GP0$, we get

$$S = Ge^{-2G}$$

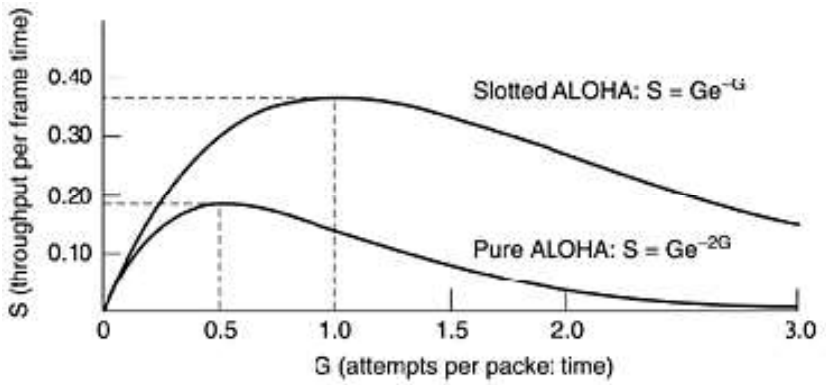
The maximum throughput occurs at $G = 0.5$, with $S = 1/2e$, which is about 0.184. In other words, the best we can hope for is a channel utilization of 18 percent.

1.1.2 Slotted ALOHA

In 1972, Roberts published a method for doubling the capacity of an ALOHA system (Roberts, 1972). His proposal was to divide time into discrete intervals, each interval corresponding to one frame. This approach requires the users to agree on slot boundaries. One way to achieve synchronization would be to have one special station emit a pip at the start of each interval, like a clock. Since the vulnerable period is now halved, the probability of no other traffic during the same slot as our test frame is *e*-*G* which leads to

$$S = Ge^{-G}$$

slotted ALOHA peaks at $G = 1$, with a throughput of $S = 1/e$ or about 0.368, twice that of pure ALOHA. If the system is operating at $G = 1$, the probability of an empty slot is 0.368



Slotted Aloha is important for a reason that may not be initially obvious. It was devised in the 1970s, used in a few early experimental systems, then almost forgotten. When Internet access over the cable was invented, all of a sudden there was a problem of how to allocate a shared channel among multiple competing users, and slotted Aloha was pulled out of the garbage can to save the day.

1.2 Carrier Sense Multiple Access Protocols

With slotted ALOHA the best channel utilization that can be achieved is 1/*e.* This is hardly surprising, since with stations transmitting at will, without paying attention to what the other stations are doing. In local area networks, it is possible for stations to detect what other stations are doing, and adapt their behavior accordingly. These networks can achieve a much better utilization than 1/*e.* Protocols in which stations listen for a carrier (i.e., a transmission) and act accordingly are called **carrier sense protocols**. A number of them have been proposed.

1.2.1 Persistent and Non-Persistent CSMA

1-persistent CSMA:

When a station has data to send, it first listens to the channel to see if anyone else is transmitting at that moment. If the channel is busy, the station waits until it becomes idle. When the station detects an idle channel, it transmits a frame. If a collision occurs, the station waits a random amount of time and starts all over again. The protocol is called 1-persistent because the station transmits with a probability of 1 when it finds the channel idle. The propagation delay has an important effect on the performance of the protocol. There is a small chance that just after a station begins sending, another station will become ready to send and sense the channel. If the first station's signal has not yet reached the second one, the latter will sense an idle

channel and will also begin sending, resulting in a collision. The longer the propagation delay, the more important this effect becomes, and the worse the performance of the protocol.

Even if the propagation delay is zero, there will still be collisions. If two stations become ready in the middle of a third station's transmission, both will wait politely until the transmission ends and then both will begin transmitting exactly simultaneously, resulting in a collision. Even so, this protocol is far better than pure ALOHA because both stations have the decency to desist from interfering with the third station's frame. Intuitively, this approach will lead to a higher performance than pure ALOHA. Exactly the same holds for slotted ALOHA.

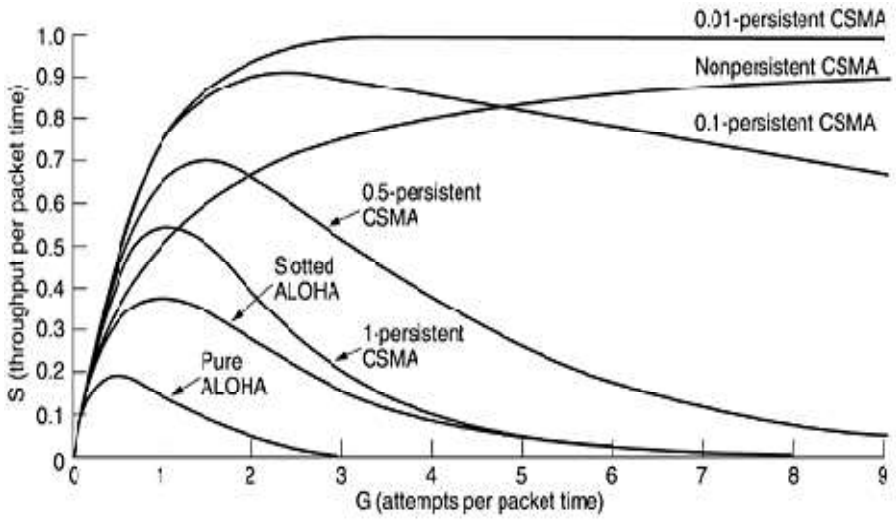
NonPersistent CSMA:

In this protocol, before sending, a station senses the channel. If no one else is sending, the station begins doing so itself. However, if the channel is already in use, the station does not continually sense it for the purpose of seizing it immediately upon detecting the end of the previous transmission. Instead, it waits a random period of time and then repeats the algorithm. Consequently, this algorithm leads to better channel utilization but longer delays than 1-persistent CSMA.

p-persistent CSMA:

It applies to slotted channels and works as follows. When a station becomes ready to send, it senses the channel. If it is idle, it transmits with a probability p. With a probability $q = 1 - p$, it defers until the next slot. If that slot is also idle, it either transmits or defers again, with probabilities p and q. This process is repeated until either the frame has been transmitted or another station has begun transmitting. In the latter case, the unlucky station acts as if there had been a collision. If the station initially senses the channel busy, it waits until the next slot and applies the above algorithm.

Comparison of the channel utilization versus load for various random access protocols.

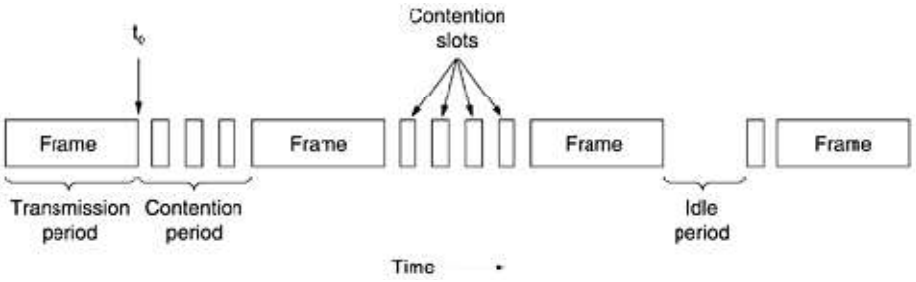


1.3 CSMA with Collision Detection

Persistent and NonPersistent CSMA protocols are clearly an improvement over ALOHA because they ensure that no station begins to transmit when it senses the channel busy. Another improvement is for stations to abort their transmissions as soon as they detect a collision. In other words, if two stations sense the channel to be idle and begin transmitting simultaneously, they will both detect the collision almost immediately. Rather than finish transmitting their frames, which are irretrievably garbled anyway, they should abruptly stop transmitting as soon as the collision is detected. Quickly terminating damaged frames saves time and bandwidth. This protocol, known as **CSMA/CD** (**CSMA with Collision Detection**) is widely used on LANs in the MAC sublayer.

At the point marked *t*0, a station has finished transmitting its frame. Any other station having a frame to send may now attempt to do so. If two or more stations decide to transmit simultaneously, there will be a collision. Collisions can be detected by looking at the power or pulse width of the received signal and comparing it to the transmitted signal.

CSMA/CD can be in one of three states: contention, transmission, or idle.



After a station detects a collision, it aborts its transmission, waits a random period of time, and then tries again, assuming that no other station has started transmitting in the meantime. Therefore, our model for CSMA/CD will consist of alternating contention and transmission periods, with idle periods occurring when all stations are quiet (e.g., for lack of work).

It is important to realize that collision detection is an *analog* process. The station's hardware must listen to the cable while it is transmitting. If what it reads back is different from what it is putting out, it knows that a collision is occurring. The implication is that the signal encoding must allow collisions to be detected.

1.4 Collision-Free Protocols

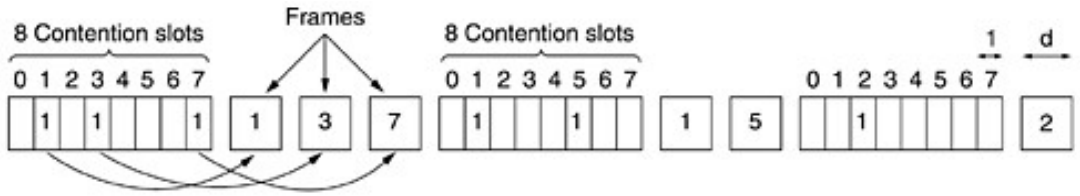
Although collisions do not occur with CSMA/CD once a station has unambiguously captured the channel, they can still occur during the contention period. These collisions adversely affect the system performance, especially when the cable is long and the frames are short. And CSMA/CD is not universally applicable. In this section, we will examine some protocols that resolve the contention for the channel without any collisions at all, not even during the contention period. Most of these are not currently used in major systems, but in a rapidly changing field, having some protocols with excellent properties available for future systems is often a good thing.

In the protocols to be described, we assume that there are exactly N stations, each with a unique address from 0 to N - 1 "wired" into it. It does not matter that some stations may be inactive part of the time. We also assume that propagation delay is negligible. The basic question remains: Which station gets the channel after a successful transmission?

1.4.1 A Bit-Map Protocol

In our first collision-free protocol, the **basic bit-map method**, each contention period consists of exactly *N* slots. If station 0 has a frame to send, it transmits a 1 bit during the zeroth slot. No other station is allowed to transmit during this slot. Regardless of what station 0 does, station 1 gets the opportunity to transmit a 1 during slot 1, but only if it has a frame queued. In general, station *j* may announce that it has a frame to send by inserting a 1 bit into slot *j*. After all *N* slots have passed by, each station has complete knowledge of which stations wish to transmit. At that point, they begin transmitting in numerical order see the following figure.

The basic bit-map protocol.



Since everyone agrees on who goes next, there will never be any collisions. After the last ready station has transmitted its frame, an event all stations can easily monitor, another *N* bit contention period is begun. If a station becomes ready just after its bit slot has passed by, it is out of luck and must remain silent until every station has had a chance and the bit map has come around again. Protocols like this in which the desire to transmit is broadcast before the actual transmission are called **reservation protocols**.

Let us briefly analyze the performance of this protocol. For convenience, we will measure time in units of the contention bit slot, with data frames consisting of *d* time units. Under conditions of low load, the bit map will simply be repeated over and over, for lack of data frames. Consider the situation from the point of view of a low-numbered station, such as 0 or 1. Typically, when it becomes ready to send, the "current" slot will be somewhere in the middle of the bit map. On average, the station will have to wait *N*/2 slots for the current scan to finish and another full *N* slots for the following scan to run to completion before it may begin transmitting. The prospects for high-numbered stations are brighter. Generally, these will only have to wait half a scan (*N*/2 bit slots) before starting to transmit. High-numbered stations rarely have to wait for the next scan. Since low-numbered stations must wait on average 1.5*N* slots and high numbered stations must wait on average 0.5*N* slots, the mean for all stations is *N* slots. The channel efficiency at low load is easy to

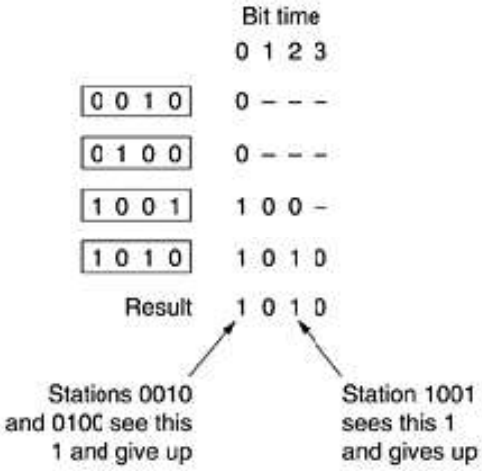
compute. The overhead per frame is *N* bits, and the amount of data is *d* bits, for an efficiency of $d/(N + d)$. At high load, when all the stations have something to send all the time, the *N* bit contention period is prorated over *N* frames, yielding an overhead of only 1 bit per frame, or an efficiency of $d/(d + 1)$.

1.4.2. Binary Countdown

A problem with the basic bit-map protocol is that the overhead is 1 bit per station, so it does not scale well to networks with thousands of stations. We can do better than that by using binary station addresses. A station wanting to use the channel now broadcasts its address as a binary bit string, starting with the high-order bit. All addresses are assumed to be the same length. The bits in each address position from different stations are BOOLEAN ORed together. We will call this protocol **binary countdown**. It was used in Datakit (Fraser, 1987). It implicitly assumes that the transmission delays are negligible so that all stations see asserted bits essentially instantaneously.

To avoid conflicts, an arbitration rule must be applied: as soon as a station sees that a highorder bit position that is 0 in its address has been overwritten with a 1, it gives up. For example, if stations 0010, 0100, 1001, and 1010 are all trying to get the channel, in the firstbit time the stations transmit 0, 0, 1, and 1, respectively. These are ORed together to form a 1. Stations 0010 and 0100 see the 1 and know that a higher-numbered station is competing for the channel, so they give up for the current round. Stations 1001 and 1010 continue. The next bit is 0, and both stations continue. The next bit is 1, so station 1001 gives up. The winner is station 1010 because it has the highest address. After winning the bidding, it may now transmit a frame, after which another bidding cycle starts. The protocol is illustrated in the following figure. It has the property that higher-numbered stations have a higher priority than lower-numbered stations, which may be either good or bad, depending on the context.

The binary countdown protocol. A dash indicates silence.



The channel efficiency of this method is $d/(d + \log 2\ N)$. If, however, the frame format has been cleverly chosen so that the sender's address is the first field in the frame, the efficiency is 100 percent. Mok and Ward (1979) have described a variation of binary countdown using a parallel rather than a serial interface. They also suggest using virtual station numbers, with the virtual station numbers from 0 up to and including the successful station being circularly permuted after each transmission, in order to give higher priority to stations that have been silent unusually long. For example, if stations *C*, *H*, *D*, *A*, *G*, *B*, *E*, *F* have priorities 7, 6, 5, 4, 3, 2, 1, and 0, respectively, then a successful transmission by *D* puts it t the end of the list, giving a priority order of *C*, *H*, *A*, *G*, *B*, *E*, *F*, *D*. Thus, *C* remains virtual station 7, but *A* moves up from 4 to 5 and *D* drops from 5 to 0. Station *D* will now only be able to acquire the channel if no other station wants it.

1.5 Limited-Contention Protocols

We have now considered two basic strategies for channel acquisition in a cable network: contention, as in CSMA, and collision-free methods. Each strategy can be rated as to how well it does with respect to the two important performance measures, delay at low load and channel efficiency at high load.

- Under conditions of light load, **contention protocols** are preferable due to its low delay. As the load increases, contention becomes increasingly less attractive,
- Just the reverse is true for the **collision-free protocols**. At low load, they have high delay, but as the load increases, the channel efficiency improves rather than gets worse as it does for contention protocols.

Obviously, it would be nice if we could combine the best properties of the contention and collision-free protocols, arriving at a new protocol that used contention at low load to provide low delay, but used a

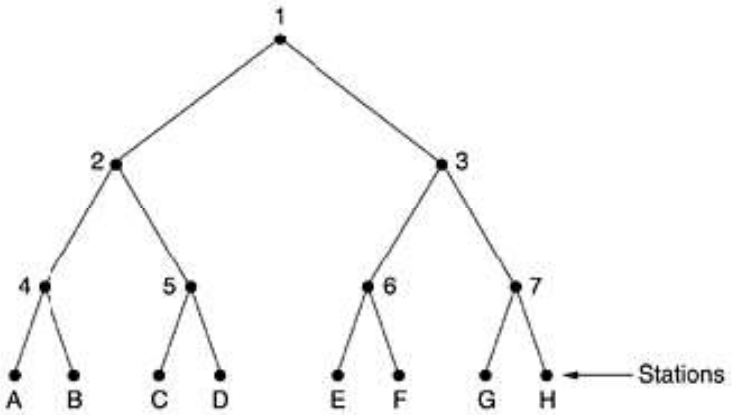
collision-free technique at high load to provide good channel efficiency. Such protocols, which we will call **limited-contention protocols**, do, in fact, exist, and will conclude our study of carrier sense networks.

The Adaptive Tree Walk Protocol

One particularly simple way of performing the necessary assignment is to use the algorithm devised by the U.S. Army for testing soldiers for syphilis during World War II (Dorfman, 1943). In short, the Army took a blood sample from *N* soldiers. A portion of each sample was poured into a single test tube. This mixed sample was then tested for antibodies. If none were found, all the soldiers in the group were declared healthy. If antibodies were present, two new mixed samples were prepared, one from soldiers 1 through *N*/2 and one from the rest. The process was repeated recursively until the infected soldiers were determined.

For the computerized version of this algorithm (Capetanakis, 1979), it is convenient to think of the stations as the leaves of a binary tree, as illustrated in the following figure. In the first contention slot following a successful frame transmission, slot 0, all stations are permitted to try to acquire the channel. If one of them does so, fine. If there is a collision, then during slot 1 only those stations falling under node 2 in the tree may compete. If one of them acquires the channel, the slot following the frame is reserved for those stations under node 3. If, on the other hand, two or more stations under node 2 want to transmit, there will be a collision during slot 1, in which case it is node 4's turn during slot 2.

The tree for eight stations.



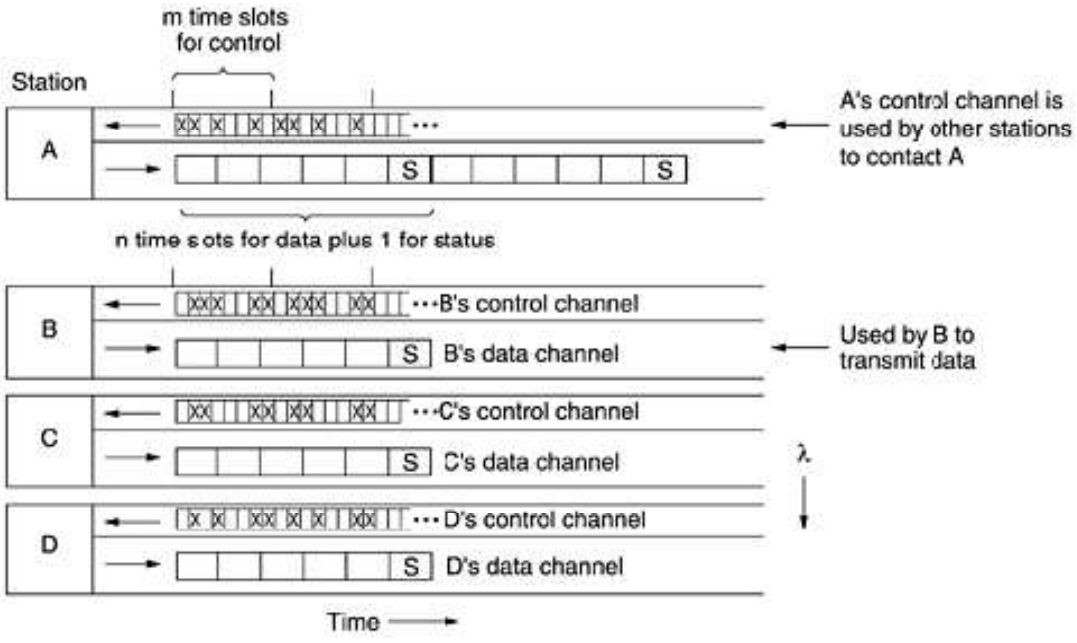
In essence, if a collision occurs during slot 0, the entire tree is searched, depth first, to locate all ready stations. Each bit slot is associated with some particular node in the tree. If a collision occurs, the search continues recursively with the node's left and right children.

4.2.5 Wavelength Division Multiple Access Protocols

A different approach to channel allocation is to divide the channel into subchannels using FDM, TDM, or both, and dynamically allocate them as needed. Schemes like this are commonly used on fiber optic LANs to permit different conversations to use different wavelengths (i.e., frequencies) at the same time. In this section we will examine one such protocol (Humblet et al., 1992). In effect, two fibers from each station are fused to a glass cylinder. One fiber is for output to the cylinder and one is for input from the cylinder. Light output by any station illuminates the cylinder and can be detected by all the other stations.

WDMA (Wavelength Division Multiple Access), each station is assigned two channels. A narrow channel is provided as a control channel to signal the station, and a wide channel is provided so the station can output data frames. Each channel is divided into groups of time slots, as shown in the following figure. Let us call the number of slots in the control channel *m* and the number of slots in the data channel *n* + 1, where *n* of these are for data and the last one is used by the station to report on its status (mainly, which slots on both channels are free). On both channels, the sequence of slots repeats endlessly, with slot 0 being marked in a special way so latecomers can detect it. All channels are synchronized by a single global clock.

Wavelength division multiple access.



The protocol supports three traffic classes :

(1) constant data rate connection-oriented traffic, such as uncompressed video,

(2) variable data rate connection-oriented traffic, such as file transfer, and
(3) datagram traffic, such as UDP packets.

For the two connection-oriented protocols, the idea is that for *A* to communicate with *B*, it must first insert a CONNECTION REQUEST frame in a free slot on *B*'s control channel. If *B* accepts, communication can take place on *A*'s data channel. Each station has two transmitters and two receivers, as follows:

1. A fixed-wavelength receiver for listening to its own control channel.
2. A tunable transmitter for sending on other stations' control channels.
3. A fixed-wavelength transmitter for outputting data frames.
4. A tunable receiver for selecting a data transmitter to listen to.

In other words, every station listens to its own control channel for incoming requests but has to tune to the transmitter's wavelength to get the data. Let us now consider how station *A* sets up a class 2 communication channel with station *B* for, say, file transfer. First, *A* tunes its data receiver to *B*'s data channel and waits for the status slot. This slot tells which control slots are currently assigned and which are free. In the above example,we see that of *B*'s eight control slots, 0, 4, and 5 are free. The rest are occupied (indicated by crosses).

A picks one of the free control slots, say, 4, and inserts its CONNECTION REQUEST message there. Since *B* constantly monitors its control channel, it sees the request and grants it by assigning slot 4 to *A*. This assignment is announced in the status slot of *B*'s data channel. When *A* sees the announcement, it knows it has a unidirectional connection. If *A* asked for a two-way connection, *B* now repeats the same algorithm with *A*. It is possible that at the same time *A* tried to grab *B*'s control slot 4, *C* did the same thing. Neither will get it, and both will notice the failure by monitoring the status slot in *B*'s control channel. They now each wait a random amount of time and try again later.

4.2.6 Wireless LAN Protocols

As the number of mobile computing and communication devices grows, so does the demand to connect them to the outside world. A common configuration for a wireless LAN is an office building with base stations (also called access points) strategically placed around the building. All the base stations are wired together using copper or fiber. If the transmission power of the base stations and notebooks is adjusted

to have a range of 3 or 4 meters, then each room becomes a single cell and the entire building becomes a large cellular system.

A naive approach to using a wireless LAN might be to try CSMA: just listen for other transmissions and only transmit if no one else is doing so. The trouble is, this protocol is not really appropriate because what matters is interference at the receiver, not at the sender. To see the nature of the problem, consider following figure, where four wireless stations are illustrated. For our purposes, it does not matter which are base stations and which are notebooks. The radio range is such that *A* and *B* are within each other's range and can potentially interfere with one another. *C* can also potentially interfere with both *B* and *D*, but not with *A*.

A wireless LAN. (a)* A *transmitting. (b)* B *transmitting.

First consider what happens when *A* is transmitting to *B*, as depicted in the above figure (a). If *C* senses the medium, it will not hear *A* because *A* is out of range, and thus falsely conclude that it can transmit to *B*. If *C* does start transmitting, it will interfere at *B*, wiping out the frame from *A*. The problem of a station not being able to detect a potential competitor for the medium because the competitor is too far away is called the **hidden station problem**.

Now let us consider the reverse situation: *B* transmitting to *A*, as shown in the above figure (b). If *C* senses the medium, it will hear an ongoing transmission and falsely conclude that it may not send to *D*, when in fact such a transmission would cause bad reception only in the zone between *B* and *C*, where neither of the intended receivers is located. This is called the **exposed station problem**.

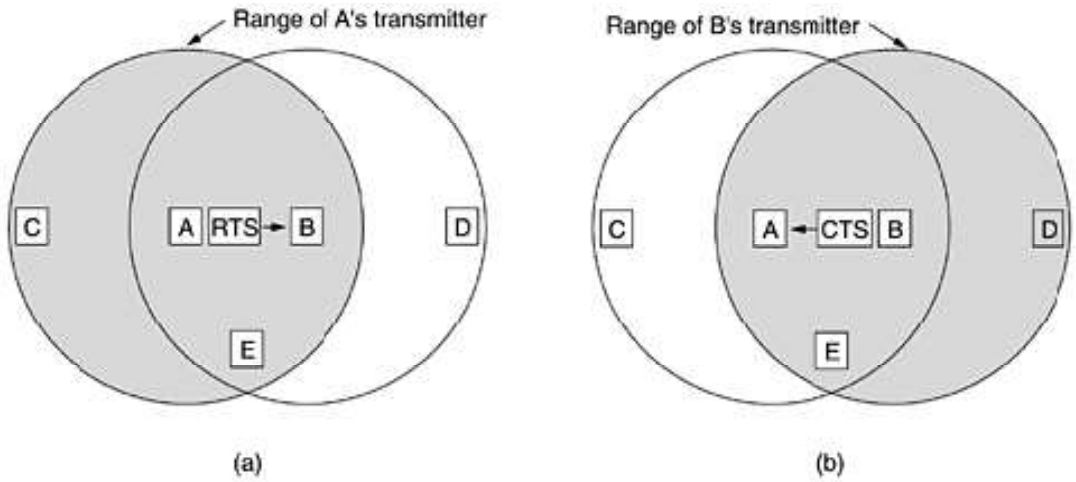
The problem is that before starting a transmission, a station really wants to know whether there is activity around the receiver. CSMA merely tells it whether there is activity around the station sensing the

carrier. Another way to think about this problem is to imagine an office building in which every employee has a wireless notebook computer. Suppose that Linda wants to send a message to Milton. Linda's computer senses the local environment and, detecting no activity, starts sending. However, there may still be a collision in Milton's office because a third party may currently be sending to him from a location so far from Linda that her computer could not detect it.

MACA and MACAW

An early protocol designed for wireless LANs is **MACA** (**Multiple Access with Collision Avoidance**) (Karn, 1990). The basic idea behind it is for the sender to stimulate the receiver into outputting a short frame, so stations nearby can detect this transmission and avoid transmitting for the duration of the upcoming (large) data frame. MACA is illustrated in the following figure

***The MACA protocol. (a)* A *sending an RTS to* B. *(b)* B *responding with a CTS to* A.**



Let us now consider how *A* sends a frame to *B*. *A* starts by sending an **RTS** (**Request To Send**) frame to *B*, as shown above. This short frame (30 bytes) contains the length of the data frame that will eventually follow. Then *B* replies with a **CTS** (**Clear to Send**) frame, as shown above. The CTS frame contains the data length (copied from the RTS frame). Upon receipt of the CTS frame, *A* begins transmission. Now let us see how stations overhearing either of these frames react. Any station hearing the RTS is clearly close to *A* and must remain silent long enough for the CTS to be transmitted back to *A* without conflict. Any station hearing the CTS is clearly close to *B* and must remain silent during the upcoming data transmission, whose length it can tell by examining the CTS frame.

In Fig. 4-12, *C* is within range of *A* but not within range of *B*. Therefore, it hears the RTS from *A* but not the CTS from *B*. As long as it does not interfere with the CTS, it is free to transmit while the data frame is being sent. In contrast, *D* is within range of *B* but not *A*. It does not hear the RTS but does hear the CTS. Hearing the CTS tips it off that it is close to a station that is about to receive a frame, so it defers sending anything until that frame is expected to be finished. Station *E* hears both control messages and, like *D*, must be silent until the data frame is complete.

5. Ethernet

The IEEE has standardized a number of local area networks and metropolitan area networks under the name of IEEE 802. A few have survived but many have not, The most important of the survivors are 802.3 (Ethernet) and 802.11 (wireless LAN). With 802.15 (Bluetooth) and 802.16 (wireless MAN). Ethernet and IEEE 802.3 are identical except for two minor differences many people use the terms "Ethernet" and "IEEE 802.3" interchangeably.

5.1 Ethernet Cabling

Since the name "Ethernet" refers to the cable (the ether), let us start our discussion there. Four types of cabling are commonly used, as shown in the following figure

The most common kinds of Ethernet cabling.

Name	Cable	Max. seg.	Nodes/seg.	Advantages
10Base5	Thick coax	500 m	100	Original cable; now obsolete
10Base2	Thin coax	185 m	30	No hub needed
10Base-T	Twisted pair	100 m	1024	Cheapest system
10Base-F	Fiber optics	2000 m	1024	Best between buildings

10Base5

Historically, **10Base5** cabling, popularly called **thick Ethernet,** came first. It resembles a yellow garden hose, with markings every 2.5 meters to show where the taps go. Connections to it are generally made using **vampire taps**, in which a pin is *very* carefully forced halfway into the coaxial cable's core. The notation 10Base5 means that it operates at 10 Mbps, uses baseband signaling, and can support segments of up to 500 meters. The first number is the speed in Mbps. Then comes the word "Base" (or sometimes "BASE") to

indicate baseband transmission. Finally, if the medium is coax, its length is given rounded to units of 100 m after "Base."

10Base2

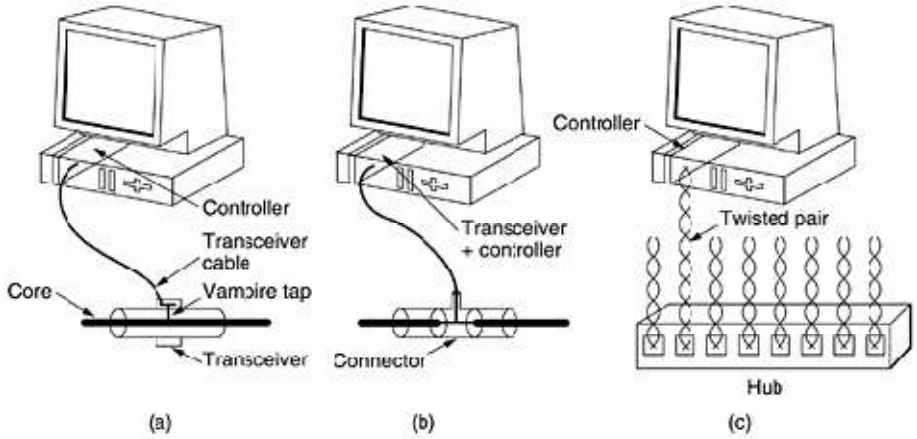
Historically, the second cable type was **10Base2**, or **thin Ethernet,** which, in contrast to the garden-hose-like thick Ethernet, bends easily. Connections to it are made using industry standard BNC connectors to form T junctions, rather than using vampire taps. BNC connectors are easier to use and more reliable. Thin Ethernet is much cheaper and easier to install, but it can run for only 185 meters per segment, each of which can handle only 30 machines. Detecting cable breaks, excessive length, bad taps, or loose connectors can be a major problem with both media. For this reason, techniques have been developed to track them down. Basically, a pulse of known shape is injected into the cable. If the pulse hits an obstacle or the end of the cable, an echo will be generated and sent back. By carefully timing the interval between sending the pulse and receiving the echo, it is possible to localize the origin of the echo. This technique is called **time domain reflectometry**.

10Base-T

A different kind of wiring pattern, in which all stations have a cable running to a central **hub** in which they are all connected electrically. Usually, these wires are telephone company twisted pairs, since most office buildings are already wired this way, and normally plenty of spare pairs are available. This scheme is called **10Base-T**. Hubs do not buffer incoming traffic.

These three **wiring schemes** are illustrated in the following figure. For 10Base5, a **transceiver** is clamped securely around the cable so that its tap makes contact with the inner core. The transceiver contains the electronics that handle carrier detection and collision detection. When a collision is detected, the transceiver also puts a special invalid signal on the cable to ensure that all other transceivers also realize that a collision has occurred.

Three kinds of Ethernet cabling. (a) 10Base5. (b)10Base2. (c) 10Base-T.



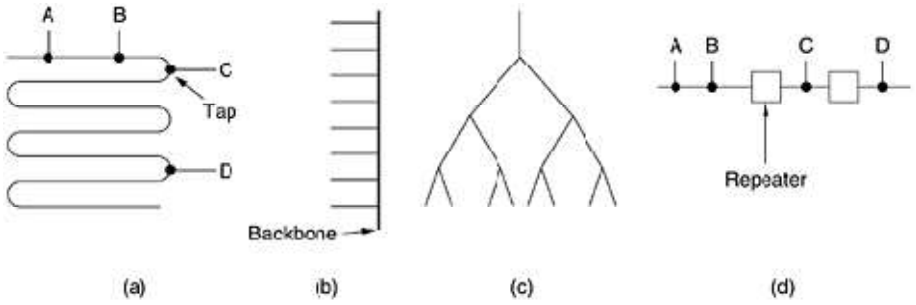
With 10Base5, a **transceiver cable** or **drop cable** connects the transceiver to an interface board in the computer. The transceiver cable may be up to 50 meters long and contains five individually shielded twisted pairs. Two of the pairs are for data in and data out, respectively. Two more are for control signals in and out. The fifth pair, which is not always used, allows the computer to power the transceiver electronics. Some transceivers allow up to eight nearby computers to be attached to them, to reduce the number of transceivers needed. The transceiver cable terminates on an interface board inside the computer. The interface board contains a controller chip that transmits frames to, and receives frames from, the transceiver. The controller is responsible for assembling the data into the proper frame format, as well as computing checksums on outgoing frames and verifying them on incoming frames. Some controller chips also manage a pool of buffers for incoming frames, a queue of buffers to be transmitted, direct memory transfers with the host computers, and other aspects of network management.

With 10Base2, the connection to the cable is just a passive BNC T-junction connector. The transceiver electronics are on the controller board, and each station always has its own transceiver. With 10Base-T, there is no shared cable at all, just the hub (a box full of electronics) to which each station is connected by a dedicated (i.e., not shared) cable. Adding or removing a station is simpler in this configuration, and cable breaks can be detected easily. The disadvantage of 10Base-T is that the maximum cable run from the hub is only 100 meters, maybe 200 meters if very high quality category 5 twisted pairs are used. Nevertheless, 10Base-T quickly became dominant due to its use of existing wiring and the ease of maintenance that it offers.

10Base-F

Uses fiber optics. This alternative is expensive due to the cost of the connectors and terminators, but it has excellent noise immunity and is the method of choice when running between buildings or widely-separated hubs. Runs of up to km are allowed. It also offers good security since wiretapping fiber is much more difficult than wiretapping copper wire. The following figure shows different ways of wiring a building.

Cable topologies. (a) Linear. (b) Spine. (c) Tree. (d) Segmented.



(a) A single cable is snaked from room to room, with each station tapping into it at the nearest point.

(b) A vertical spine runs from the basement to the roof, with horizontal cables on each floor connected to the spine by special amplifiers (repeaters). In some buildings, the horizontal cables are thin and the backbone is thick.

(c) The most general topology is the tree, because a network with two paths between some pairs of stations would suffer from interference between the two signals.

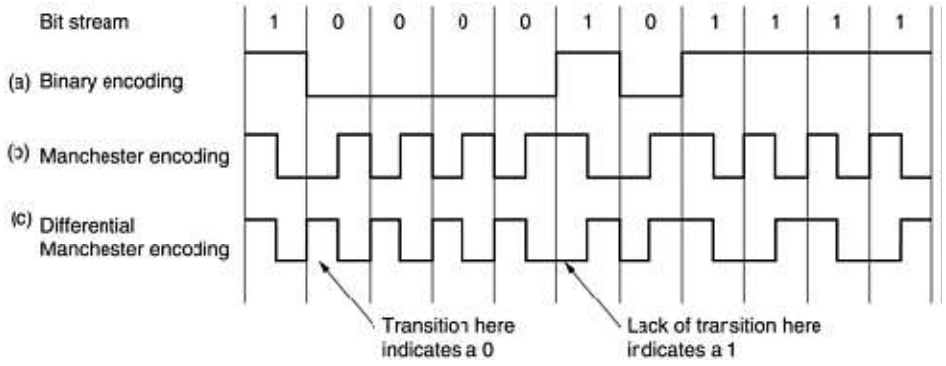
(d) Each version of Ethernet has a maximum cable length per segment. To allow larger networks, multiple cables can be connected by **repeaters**, as shown in (d). A repeater is a physical layer device. It receives, amplifies (regenerates), and retransmits signals in both directions. As far as the software is concerned, a series of cable segments connected by repeaters is no different from a single cable. A system may contain multiple cable segments and multiple repeaters

5.2 Manchester Encoding

None of the versions of Ethernet uses straight binary encoding with 0 volts for a 0 bit and 5 volts for a 1 bit because it leads to ambiguities. If one station sends the bit string 0001000, others might falsely interpret it as 10000000 or 01000000 because they cannot tell the difference between an idle sender (0 volts)

and a 0 bit (0 volts). This problem can be solved by using +1 volts for a 1 and -1 volts for a 0, but there is still the problem of a receiver sampling the signal at a slightly different frequency than the sender used to generate it. Different clock speeds can cause the receiver and sender to get out of synchronization about where the bit boundaries are, especially after a long run of consecutive 0s or a long run of consecutive 1s. What is needed is a way for receivers to unambiguously determine the start, end, or middle of each bit without reference to an external clock. Two such approaches are called **Manchester encoding** and **differential Manchester encoding**. With Manchester encoding, each bit period is divided into two equal intervals. A binary 1 bit is sent by having the voltage set high during the first interval and low in the second one. A binary 0 is just the reverse: first low and then high. This scheme ensures that every bit period has a transition in the middle, making it easy for the receiver to synchronize with the sender. A disadvantage of Manchester encoding is that it requires twice as much bandwidth as straight binary encoding because the pulses are half the width. For example, to send data at 10 Mbps, the signal has to change 20 million times/sec. Manchester encoding is shown below

(a) Binary encoding. (b) Manchester encoding. (c) Differential Manchester encoding.

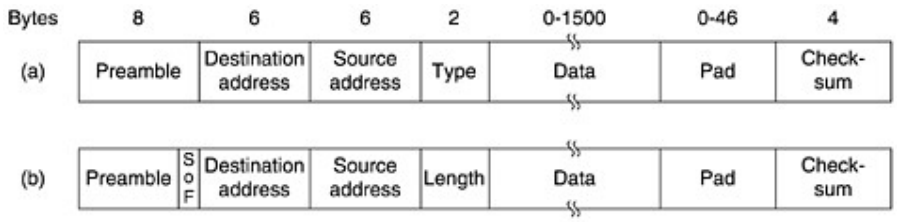


Differential Manchester encoding is a variation of basic Manchester encoding. In it, a 1 bit is indicated by the absence of a transition at the start of the interval. A 0 bit is indicated by the presence of a transition at the start of the interval. In both cases, there is a transition in the middle as well. The differential scheme requires more complex equipment but offers better noise immunity. All Ethernet systems use Manchester encoding due to its simplicity. The high signal is + 0.85 volts and the low signal is - 0.85 volts, giving a DC value of 0 volts. Ethernet does not use differential Manchester encoding, but other LANs (e.g., the 802.5 token ring) do use it.

5.3 The Ethernet MAC Sublayer Protocol

The original DIX (DEC, Intel, Xerox) frame structure is shown in figure (a). Each frame starts with a ***Preamble*** of 8 bytes, each containing the bit pattern 10101010. The Manchester encoding of this pattern produces a 10-MHz square wave for 6.4 μsec to allow the receiver's clock to synchronize with the sender's. They are required to stay synchronized for the rest of the frame, using the Manchester encoding to keep track of the bit boundaries.

Frame formats. (a) DIX Ethernet. (b) IEEE 802.3.



The frame contains two addresses, one for the **destination** and one for the **source**. The high-order bit of the destination address is a 0 for ordinary addresses and 1 for group addresses. Group addresses allow multiple stations to listen to a single address. When a frame is sent to a group address, all the stations in the group receive it. Sending to a group of stations is called **multicast**. The address consisting of all 1 bits is reserved for **broadcast**.

Another interesting feature of the addressing is the use of bit 46 (adjacent to the high-order bit) to distinguish local from global addresses. Next comes the ***Type*** field, which tells the receiver what to do with the frame. Multiple network-layer protocols may be in use at the same time on the same machine, so when an Ethernet frame arrives, the kernel has to know which one to hand the frame to. The *Type* field specifies which process to give the frame to. Next come the **data**, up to 1500 bytes. Ethernet requires that valid frames must be at least 64 bytes long, from destination address to checksum, including both. If the data portion of a frame is less than 46 bytes, the ***Pad*** field is used to fill out the frame to the minimum size. Frames with fewer than 64 bytes are padded out to 64 bytes with the *Pad* field. The final Ethernet field is the ***Checksum***. It is effectively a 32-bit hash code of the data. If some data bits are erroneously received (due to noise on the cable), the checksum will almost certainly be wrong and the error will be detected. The

checksum algorithm is a cyclic redundancy check (CRC) of the kind discussed in Chap. 3. It just does error detection, not forward error correction.

When IEEE standardized Ethernet, the committee made two changes to the DIX format, as shown in FIGURE (b). The first one was to reduce the preamble to 7 bytes and use the last byte for a ***Start of Frame (STF)*** delimiter, for compatibility with 802.4 and 802.5. The second one was to change the ***Type*** field into a ***Length*** field. Of course, now there was no way for the receiver to figure out what to do with an incoming frame, but that problem was handled by the addition of a small header to the data portion itself to provide this information.

5.4. The Binary Exponential Backoff Algorithm

Let us now see how randomization is done when a collision occurs. After a collision, time is divided into discrete slots whose length is equal to the worst-case round-trip propagation time on the ether (2τ). To accommodate the longest path allowed by Ethernet, the slot time has been set to 512 bit times, or 51.2 μsec as mentioned above. After the first collision, each station waits either 0 or 1 slot times before trying again. If two stations collide and each one picks the same random number, they will collide again. After the second collision, each one picks either 0, 1, 2, or 3 at random and waits that number of slot times. If a third collision occurs (the probability of this happening is 0.25), then the next time the number of slots to wait is chosen at random from the interval 0 to 23 - 1.

In general, after i collisions, a random number between 0 and $2i$ - 1 is chosen, and that number of slots is skipped. However, after ten collisions have been reached, the randomization interval is frozen at a maximum of 1023 slots. This algorithm, called **binary exponential backoff**, was chosen to dynamically adapt to the number of stations trying to send.

5.5. Ethernet Performance

Now let us briefly examine the performance of Ethernet under conditions of heavy and constant load, that is, k stations always ready to transmit. A rigorous analysis of the binary exponential backoff algorithm is complicated. If each station transmits during a contention slot with probability p, the probability A that some station acquires the channel in that slot is

$$A = kp(1-p)^{k-1}$$

A is maximized when $p = 1/k$, with A $1/e$ as k . The probability that the contention interval has exactly j slots in it is $A(1 - A)j - 1$, so the mean number of slots per contention is given by

$$\sum_{j=0}^{\infty} jA(1-A)^{j-1} = \frac{1}{A}$$

Since each slot has a duration 2τ, the mean contention interval, w, is $2\tau/A$. Assuming optimal p, the mean number of contention slots is never more than e, so w is at most $2\tau e$ 5.4τ. If the mean frame takes P sec to transmit, when many stations have frames to send,

$$\text{Channel efficiency} = \frac{P}{P + 2\tau/A}$$

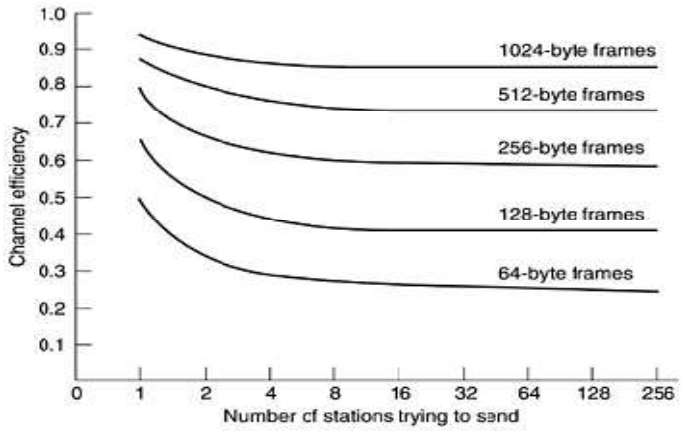
The longer the cable, the longer the contention interval. This observation is why the Ethernet standard specifies a maximum cable length.

The formulae in terms of the frame length, F, the network bandwidth, B, the cable length, L, and the speed of signal propagation, c, for the optimal case of e contention slots per frame. With $P = F/B$,

$$\text{Channel efficiency} = \frac{1}{1 + 2BLe/cF}$$

When the second term in the denominator is large, network efficiency will be low. More specifically, increasing network bandwidth or distance (the BL product) reduces efficiency for a given frame size.

Efficiency of Ethernet at 10 Mbps with 512-bit slot times



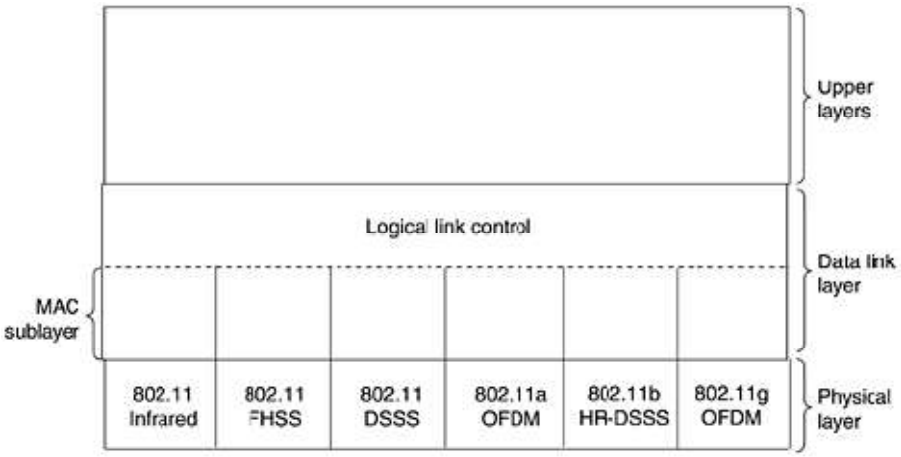
6. Wireless LANs

Wireless LANs are increasingly popular, and more and more office buildings, airports, and other public places are being outfitted with them. Wireless LANs can operate in one of two configurations, with a base station and without a base station. Consequently, the 802.11 LAN standard takes this into account and makes provision for both arrangements.

6.1 The 802.11 Protocol Stack

The protocols used by all the 802 variants, including Ethernet, have a certain commonality of structure. A partial view of the 802.11 protocol stack is given in following. The physical layer corresponds to the OSI physical layer fairly well, but the data link layer in all the 802 protocols is split into two or more sublayers. In 802.11, the MAC (Medium Access Control) sublayer determines how the channel is allocated, that is, who gets to transmit next. Above it is the LLC (Logical Link Control) sublayer, whose job it is to hide the differences between the different 802 variants and make them indistinguishable as far as the network layer is concerned.

Part of the 802.11 protocol stack.



The 1997 802.11 standard specifies three transmission techniques allowed in the physical layer. The **infrared method** uses much the same technology as television remote controls do. The other two use short-range radio, using techniques called **FHSS and DSSS**. Both of these use a part of the spectrum that does not require licensing (the 2.4-GHz ISM band). All of these techniques operate at 1 or 2 Mbps and at low enough power that they do not conflict too much. In 1999, two new techniques were introduced to achieve higher

bandwidth. These are called **OFDM and HR-DSSS**. They operate at up to 54 Mbps and 11 Mbps, respectively. In 2001, a second **OFDM modulation** was introduced, but in a different frequency band from the first one. Now we will examine each of them briefly. Technically, these belong to the physical, but since they are so closely tied to LANs in general and the 802.11 MAC sublayer, we treat them here instead.

6.2 The 802.11 Physical Layer

Each of the five permitted transmission techniques makes it possible to send a MAC frame from one station to another. They differ, however, in the technology used and speeds achievable.

Infrared

This option uses diffused (i.e., not line of sight) transmission at 0.85 or 0.95 microns. Two speeds are permitted: 1 Mbps and 2 Mbps. At 1 Mbps, an encoding scheme is used in which a group of 4 bits is encoded as a 16-bit codeword containing fifteen 0s and a single 1, using what is called **Gray code**. This code has the property that a small error in time synchronization leads to only a single bit error in the output. Infrared signals cannot penetrate walls, so cells in different rooms are well isolated from each other. Nevertheless, due to the low bandwidth this is not a popular option.

FHSS (Frequency Hopping Spread Spectrum)

Uses 79 channels, each 1-MHz wide, starting at the low end of the 2.4-GHz ISM band. A pseudorandom number generator is used to produce the sequence of frequencies hopped to. As long as all stations use the same seed to the pseudorandom number generator and stay synchronized in time, they will hop to the same frequencies simultaneously. The amount of time spent at each frequency, the **dwell time**, is an adjustable parameter, but must be less than 400 msec. FHSS' randomization provides a fair way to allocate spectrum in the unregulated ISM band. It also provides a modicum of security since an intruder who does not know the hopping sequence or dwell time cannot eavesdrop on transmissions. Over longer distances, multipath fading can be an issue, and FHSS offers good resistance to it. It is also relatively insensitive to radio interference, which makes it popular for building-to-building links. Its main disadvantage is its low bandwidth.

DSSS (Direct Sequence Spread Spectrum)

It is also restricted to 1 or 2 Mbps. The scheme used has some similarities to the CDMA system but differs in other ways. Each bit is transmitted as 11 chips, using what is called a **Barker sequence**. It uses

phase shift modulation at 1 Mbaud, transmitting 1 bit per baud when operating at 1 Mbps and 2 bits per baud when operating at 2 Mbps. For years, the FCC required all wireless communications equipment operating in the ISM bands in the U.S. to use spread spectrum

OFDM (Orthogonal Frequency Division Multiplexing)

The first of the high-speed wireless LANs, **802.11a**, uses this to deliver up to 54 Mbps in the wider 5-GHz ISM band. This technique emerged in May 2002. As the term FDM suggests, different frequencies are used—52 of them, 48 for data and 4 for synchronization, transmissions are present on multiple frequencies at the same time, this technique is considered a form of spread spectrum, but different from both CDMA and FHSS. Splitting the signal into many narrow bands has some key advantages over using a single wide band, including better immunity to narrowband interference and the possibility of using noncontiguous bands. A complex encoding system is used, based on phase shift modulation for speeds up to 18 Mbps and on QAM above that. At 54 Mbps, 216 data bits are encoded into 288-bit symbols. Part of the motivation for OFDM is compatibility with the European HiperLAN/2 system (Doufexi et al., 2002). The technique has a good spectrum efficiency in terms of bits/Hz and good immunity to multipath fading.

HR-DSSS (High Rate Direct Sequence Spread Spectrum),

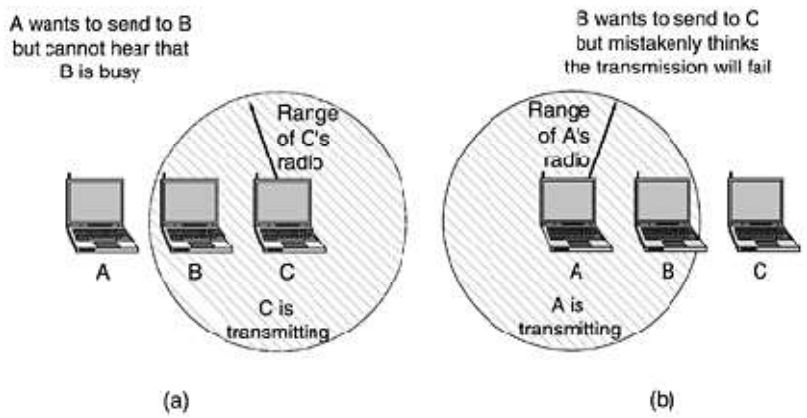
Uses 11 million chips/sec to achieve 11 Mbps in the 2.4-GHz band. It is called **802.11b** but is not a follow-up to 802.11a. In fact, its standard was approved first and it got to market first. Data rates supported by 802.11b are 1, 2, 5.5, and 11 Mbps. The two slow rates run at 1 Mbaud, with 1 and 2 bits per baud, respectively, using phase shift modulation (for compatibility with DSSS). The two faster rates run at 1.375 M baud, with 4 and 8 bits per baud, respectively, using **Walsh/ Hadamard** codes. The data rate may be dynamically adapted during operation to achieve the optimum speed possible under current conditions of load and noise. In practice, the operating speed of 802.11b is nearly always 11 Mbps. Although 802.11b is slower than 802.11a, its range is about 7 times greater, which is more important in many situations.

An enhanced version of 802.11b, **802.11g**, was approved by IEEE in November 2001 after much politicking about whose patented technology it would use. It uses the OFDM modulation method of 802.11a but operates in the narrow 2.4-GHz ISM band along with 802.11b. What it does mean is that the 802.11 committee has produced three different high speed wireless LANs: 802.11a, 802.11b, and 802.11g (not to mention three low-speed wireless LANs).

6.3. The 802.11 MAC Sublayer Protocol

The 802.11 MAC sublayer protocol is quite different from that of Ethernet due to the inherent complexity of the wireless environment compared to that of a wired system. With Ethernet, a station just waits until the ether goes silent and starts transmitting. If it does not receive a noise burst back within the first 64 bytes, the frame has almost assuredly been delivered correctly. With wireless, this situation does not hold. To start with, there is the **hidden station problem** and is illustrated in Fig. (a).

Figure (a) The hidden station problem. (b) The exposed station problem.



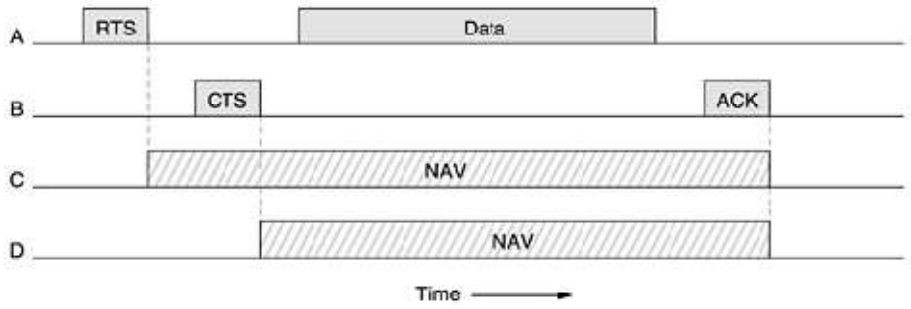
the **exposed station problem**, illustrated in Fig. (b). In addition, most radios are half duplex, meaning that they cannot transmit and listen for noise bursts at the same time on a single frequency. As a result of these problems, 802.11 does not use CSMA/CD, as Ethernet does. To deal with this problem, 802.11 supports two modes of operation.

1. **PCF** (**Point Coordination Function**), uses the base station to control all activity in its cell.
2. **DCF** (**Distributed Coordination Function**), does not use any kind of central control. DCF is employed, 802.11 uses a protocol called **CSMA/CA** (**CSMA with Collision Avoidance**). In this protocol, both physical channel sensing and virtual channel sensing are used. Two methods of operation are supported by CSMA/CA.
    - In the first method, when a station wants to transmit, it senses the channel. If it is idle, it just starts transmitting. It does not sense the channel while transmitting but emits its entire frame, which may well be destroyed at the receiver due to interference there. If the channel is busy, the sender defers until it goes idle and then starts transmitting. If a collision occurs, the

colliding stations wait a random time, using the Ethernet binary exponential backoff algorithm, and then try again later.

- The other mode of CSMA/CA operation is based on MACAW and uses virtual channel sensing, as illustrated in the following figure. In this example, *A* wants to send to *B*. C is a station within range of *A* (and possibly within range of *B*, but that does not matter). *D* is a station within range of *B* but not within range of *A*.

Figure. The use of virtual channel sensing using CSMA/CA.

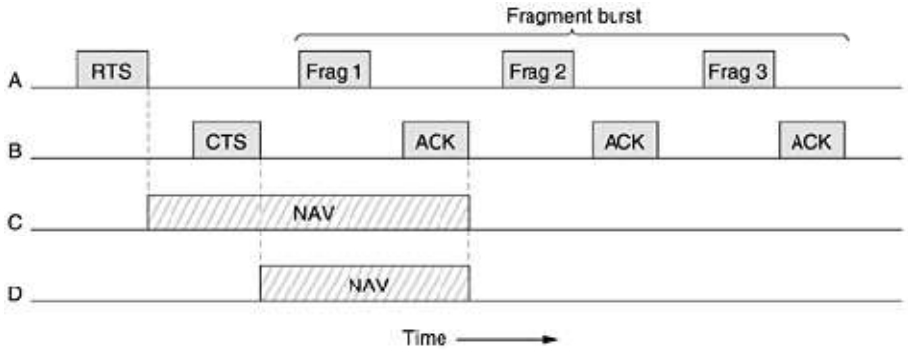


The protocol starts when *A* decides it wants to send data to *B*. It begins by sending an RTS frame to *B* to request permission to send it a frame. When *B* receives this request, it may decide to grant permission, in which case it sends a CTS frame back. Upon receipt of the CTS, *A* now sends its frame and starts an ACK timer. Upon correct receipt of the data frame, *B* responds with an ACK frame, terminating the exchange. If *A*'s ACK timer expires before the ACK gets back to it, the whole protocol is run again. Now let us consider this exchange from the viewpoints of *C* and *D*. *C* is within range of *A*, so it may receive the RTS frame. If it does, it realizes that someone is going to send data soon, so for the good of all it desists from transmitting anything until the exchange is completed.

From the information provided in the RTS request, it can estimate how long the sequence will take, including the final ACK, so it asserts a kind of virtual channel busy for itself, indicated by **NAV** (**Network Allocation Vector**) in the above figure. *D* does not hear the RTS, but it does hear the CTS, so it also asserts the *NAV* signal for itself. Note that the *NAV* signals are not transmitted; they are just internal reminders to keep quiet for a certain period of time.

To deal with the problem of noisy channels, 802.11 allows frames to be fragmented into smaller pieces, each with its own checksum. The fragments are individually numbered and acknowledged using a stop-and-wait protocol. Once the channel has been acquired using RTS and CTS, multiple fragments can be sent in a row, as shown in the following figure..
sequence of fragments is called a **fragment burst**.

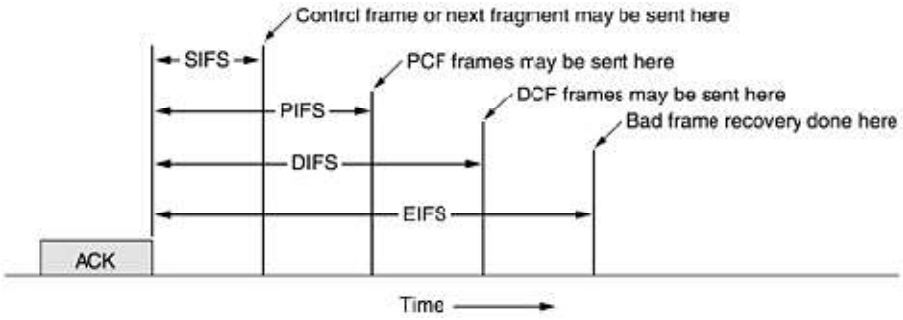
Figure . A fragment burst.



Fragmentation increases the throughput by restricting retransmissions to the bad fragments rather than the entire frame. The basic mechanism is for the base station to broadcast a **beacon frame** periodically (10 to 100 times per second). The beacon frame contains system parameters, such as hopping sequences and dwell times (for FHSS), clock synchronization, etc. It also invites new stations to sign up for polling service.

PCF and DCF can coexist within one cell. At first it might seem impossible to have central control and distributed control operating at the same time, but 802.11 provides a way to achieve this goal. It works by carefully defining the interframe time interval. After a frame has been sent, a certain amount of dead time is required before any station may send a frame. Four different intervals are defined, each for a specific purpose. The four intervals are depicted in the following figure.

Figure. Interframe spacing in 802.11



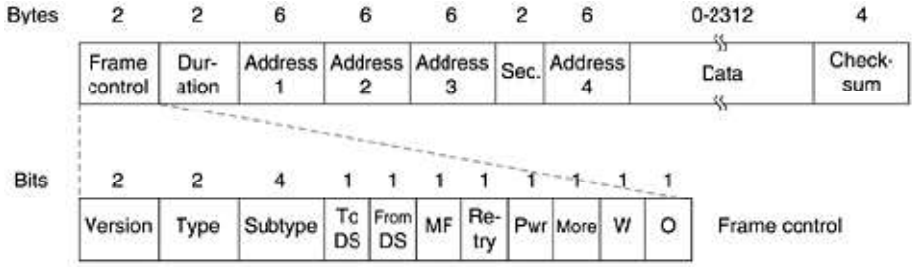
- The shortest interval is **SIFS** (**Short InterFrame Spacing**). It is used to allow the parties in a single dialog the chance to go first. This includes letting the receiver send a CTS to respond to an RTS, letting the receiver send an ACK for a fragment or full data frame, and letting the sender of a fragment burst transmit the next fragment without having to send an RTS again. There is always exactly one station that is entitled to respond after a SIFS interval.
- **PIFS** (**PCF InterFrame Spacing**), the base station may send a beacon frame or poll frame. This mechanism allows a station sending a data frame or fragment sequence to finish its frame without anyone else getting in the way, but gives the base station a chance to grab the channel when the previous sender is done without having to compete with eager users.
- If the base station has nothing to say and a time **DIFS** (**DCF InterFrame Spacing**) elapses, any station may attempt to acquire the channel to send a new frame. The usual contention rules apply, and binary exponential backoff may be needed if a collision occurs.
- The last time interval, **EIFS** (**Extended InterFrame Spacing**), is used only by a station that has just received a bad or unknown frame to report the bad frame. The idea of giving this event the lowest priority is that since the receiver may have no idea of what is going on, it should wait a substantial time to avoid interfering with an ongoing dialog between two stations.

6. 4 The 802.11 Frame Structure

The 802.11 standard defines three different classes of frames on the wire: **data, control, and management.** Each of these has a header with a variety of fields used within the MAC sublayer. The format of the **data frame** is shown in the following figure. First comes the ***Frame Control*** field. It itself has 11 subfields. The first of these is the ***Protocol version*****,** which allows two versions of the protocol to operate at

the same time in the same cell. Then come the ***Type*** (data, control, or management) and ***Subtype*** fields (e.g., RTS or CTS). The *To DS* and *From DS* bits indicate the frame is going to or coming from the intercell distribution system (e.g., Ethernet). The *MF* bit means that more fragments will follow. The ***Retry*** bit marks a retransmission of a frame sent earlier. The ***Power management*** bit is used by the base station to put the receiver into sleep state or take it out of sleep state. The ***More*** bit indicates that the sender has additional frames for the receiver. The *W* bit specifies that the frame body has been encrypted using the **WEP** (**Wired Equivalent Privacy**) algorithm. Finally, the *O* bit tells the receiver that a sequence of frames with this bit on must be processed strictly in order.

Figure . The 802.11 data frame.



The second field of the data frame, the ***Duration*** field, tells how long the frame and its acknowledgement will occupy the channel. This field is also present in the control frames and is how other stations manage the NAV mechanism. The frame header contains four addresses, all in standard IEEE 802 format. The source and destination are obviously needed, The other two addresses are used for the source and destination base stations for intercell traffic. The ***Sequence*** field allows fragments to be numbered. Of the 16 bits available, 12 identify the frame and 4 identify the fragment. The ***Data*** field contains the payload, up to 2312 bytes, followed by the usual ***Checksum.***

Management frames have a format similar to that of data frames, except without one of the base station addresses, because management frames are restricted to a single cell. Control frames are shorter still, having only one or two addresses, no ***Data*** field, and no ***Sequence*** field. The key information here is in the ***Subtype*** field, usually RTS, CTS, or ACK.

6.5 Services

The 802.11 standard states that each conformant wireless LAN must provide nine services. These services are divided into two categories: five **distribution services** and four **station services**.

The **distribution services:** relate to managing cell membership and interacting with stations outside the cell. In contrast, the station services relate to activity within a single cell. The five distribution services are provided by the base stations and deal with station mobility as they enter and leave cells, attaching themselves to and detaching themselves from base stations. They are as follows.

1. **Association.** This service is used by mobile stations to connect themselves to base stations. Typically, it is used just after a station moves within the radio range of the base station. Upon arrival, it announces its identity and capabilities. The capabilities include the data rates supported, need for PCF services (i.e., polling), and power management requirements. The base station may accept or reject the mobile station. If the mobile station is accepted, it must then authenticate itself.
2. **Disassociation.** Either the station or the base station may disassociate, thus breaking the relationship. A station should use this service before shutting down or leaving, but the base station may also use it before going down for maintenance.
3. **Reassociation.** A station may change its preferred base station using this service. This facility is useful for mobile stations moving from one cell to another. If it is used correctly, no data will be lost as a consequence of the handover. (But 802.11, like Ethernet, is just a best-efforts service.)
4. **Distribution.** This service determines how to route frames sent to the base station. If the destination is local to the base station, the frames can be sent out directly over the air. Otherwise, they will have to be forwarded over the wired network.
5. **Integration.** If a frame needs to be sent through a non-802.11 network with a different addressing scheme or frame format, this service handles the translation from the 802.11 format to the format required by the destination network.

The remaining four services are intracell. They are used after association has taken place and are as follows.

1. **Authentication.** Because wireless communication can easily be sent or received by unauthorized stations, a station must authenticate itself before it is permitted to send data. After a mobile station has been associated by the base station (i.e., accepted into its cell),
2. **Deauthentication.** When a previously authenticated station wants to leave the network, it is deauthenticated. After deauthentication, it may no longer use the network.

3. **Privacy.** For information sent over a wireless LAN to be kept confidential, it must be encrypted. This service manages the encryption and decryption. The encryption algorithm specified is RC4, invented by Ronald Rivest of M.I.T.
4. **Data delivery.** Finally, data transmission is what it is all about, so 802.11 naturally provides a way to transmit and receive data. Since 802.11 is modeled on Ethernet and transmission over Ethernet is not guaranteed to be 100% reliable, transmission over 802.11 is not guaranteed to be reliable either. Higher layers must deal with detecting and correcting errors.